(continued from Part 8)

Printers

The printer is one of the most widely used I/O peripherals. It can be used at any time to record the results of data processing so that they can be kept in an instantly readable form, which is known as a **hard copy.** A printer is essential for most business applications. Imagine a system programmed to perform a customer accounts function and not being able to print out invoices, or a word processing system which was unable to print letters.

It may be that in future, with the so-called 'paperless office' and electronic mail, hard copies will become less important, but for now printers hold a crucial role in the computer system.

Printers can use single sheets, like typewriters, or continuous forms which are pleated and then separated into individual sheets after printing. Continuous stationery – the most common type used by computer printers – has a perforated strip on each side of the paper. These strips have holes punched in them which allow the paper to be hooked onto the printer's paper transport machanism. The paper is pulled through the printer like film through a camera. There are print units available which allow both sorts of paper to feed simultaneously if necessary. You may for instance, want to interrupt a continuous print run in order to get a one-off print of a particular page.

Line spacing – usually 1/6 or 1/8 of an inch – can be chosen using a special switch. The number of characters per line varies according to the machine used. The most common character widths are 32, 80, 120 and 160 characters. If your VDU screen can display a 160 character width it is important to have a printer which can do the same, if you wish to obtain the same layout on a hard copy.

There are various ways of controlling the print layout and format. The length and number of lines on each page can either be set manually – by pressing buttons – or by using a system of preprogramming which can cope with complex printing tasks, like printing bills or scientific results and tables. *Figure 3* shows part of a punched tape from a printer control mechanism. High level programming languages have their own direct printer control instructions.

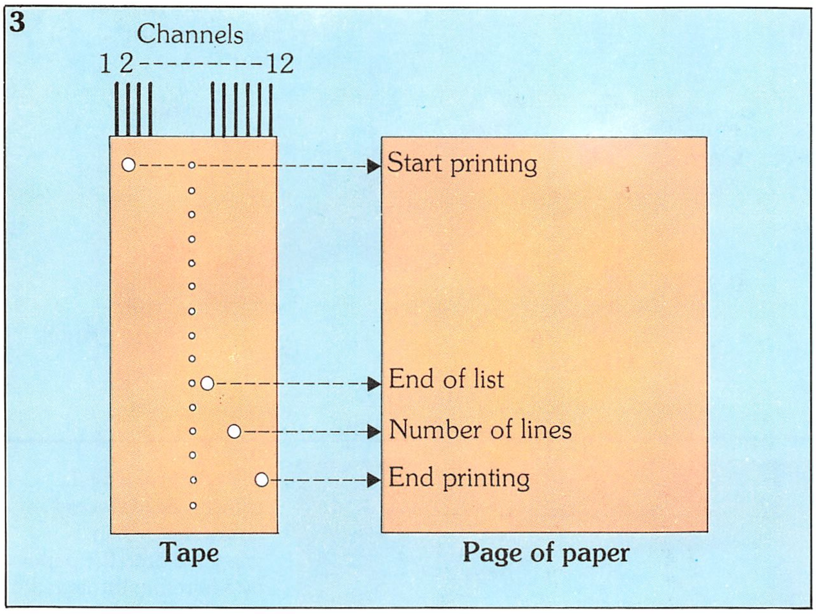
Printing methods can be divided into two broad groups: **impact** and **non-impact.** Impact printers make mechanical contact with the paper, using an inking ribbon like a typewriter's. Non-impact printing is carried out by thermal, laser or ink-jet systems.

Printers can also be subdivided into **serial** and **parallel** types. Serial printers, as the name suggests, print along the line, character by character. There are two basic sorts: **unidirectional,** which print from left to right and have a carriage return, like a typewriter, and **bidirectional,** which print

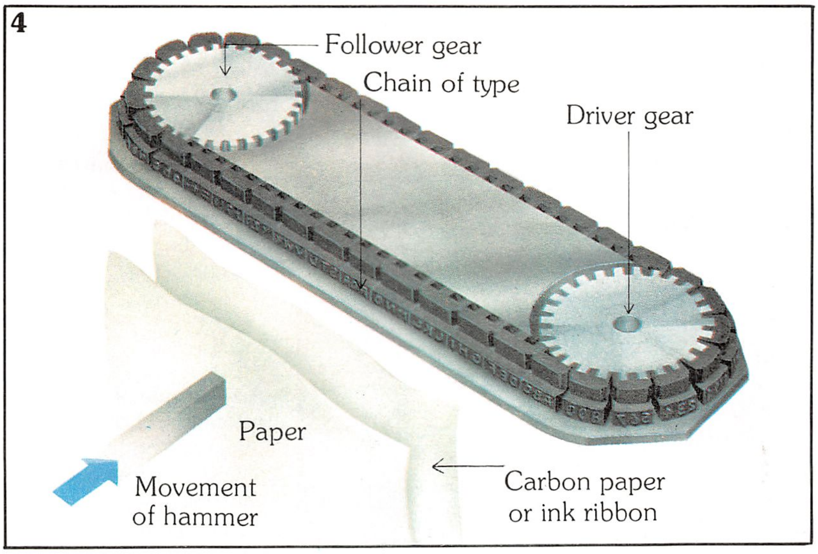
3. Part of a punched paper tape from a printer control mechanism showing how instructions are coded into the holes.

4. A chain printer. The type is carried on a continuous metal chain.

3



4



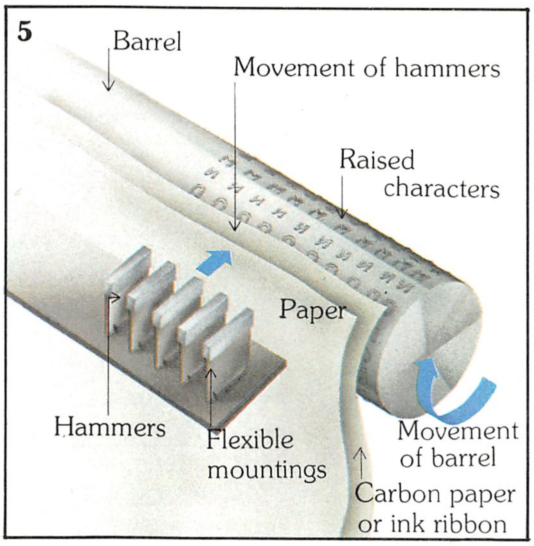
in both directions and so have no carriage return mechanism.

Parallel printers produce all the characters in a line simultaneously, or in such a way that the printing *appears* to take place simultaneously for all the character positions.

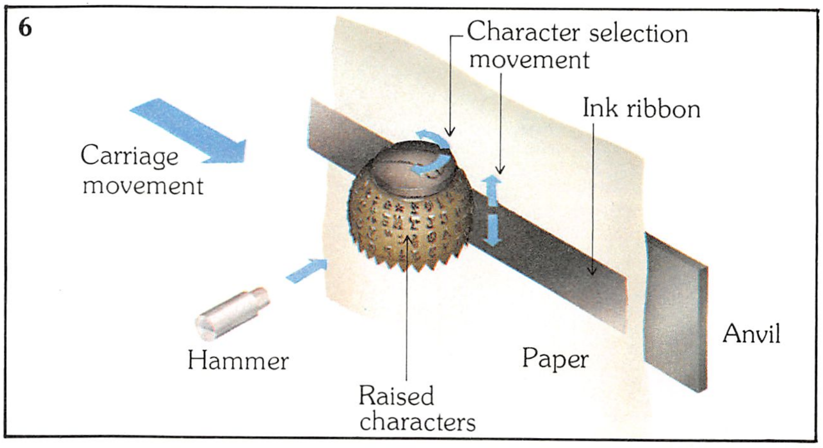
Impact printers

Large computer systems usually employ one of two types: a chain printer or a barrel printer. Chain printers, as the name implies, carry the type on a continuous metal chain (*figure* 4). Barrel printers carry the type on a rotating cylinder. Each line of type on the cylinder is made up of one individual character, and there are as many lines as there are characters in the computer's character set (*figure* 5).

Both types work on the same basic principles: the chain or barrel is continuously driven at high speed. There are print hammers for each character position in a line. A hammer presses the paper and inking ribbon against the type as the correct character passes its position. Both of these printers are parallel types and are known as **line printers.** They can print lines with up to 160 characters on continuous stationery, at speeds from about 300 to 3000 lines a minute.



5. Barrel printers carry the type on a rotating cylinder. Each line of type is made up of one individual character.

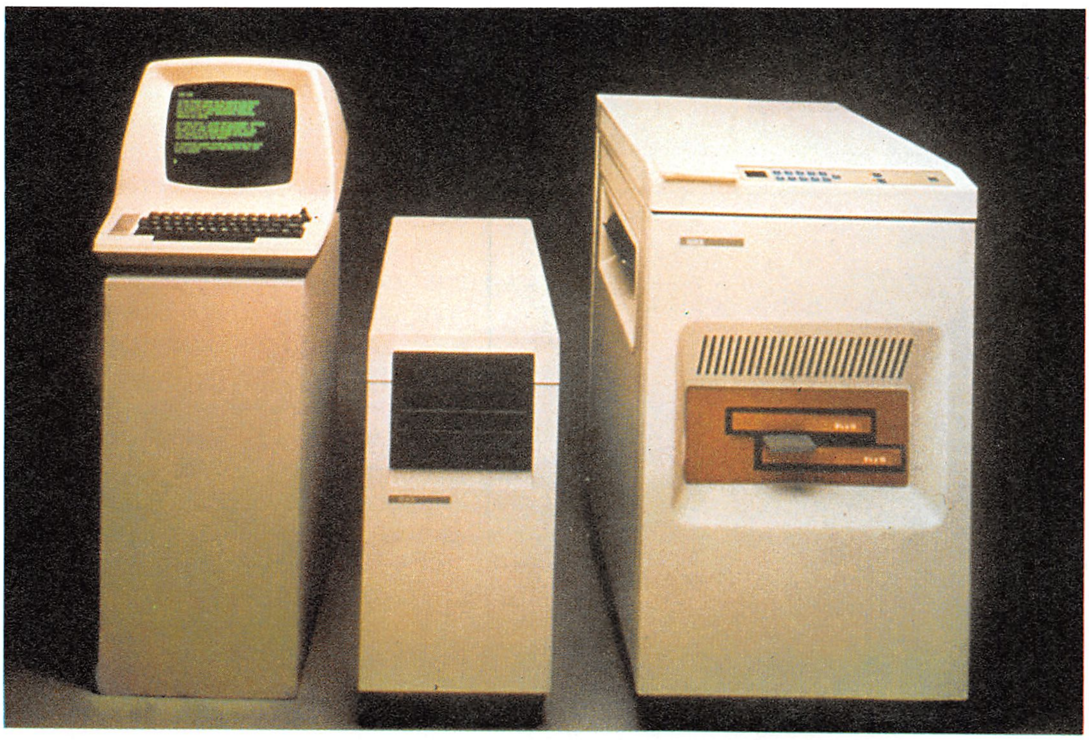


6. The golf ball printing mechanism has been extensively used in office typewriters for many years.

7. Example of a daisy wheel. These can either be metal or plastic.

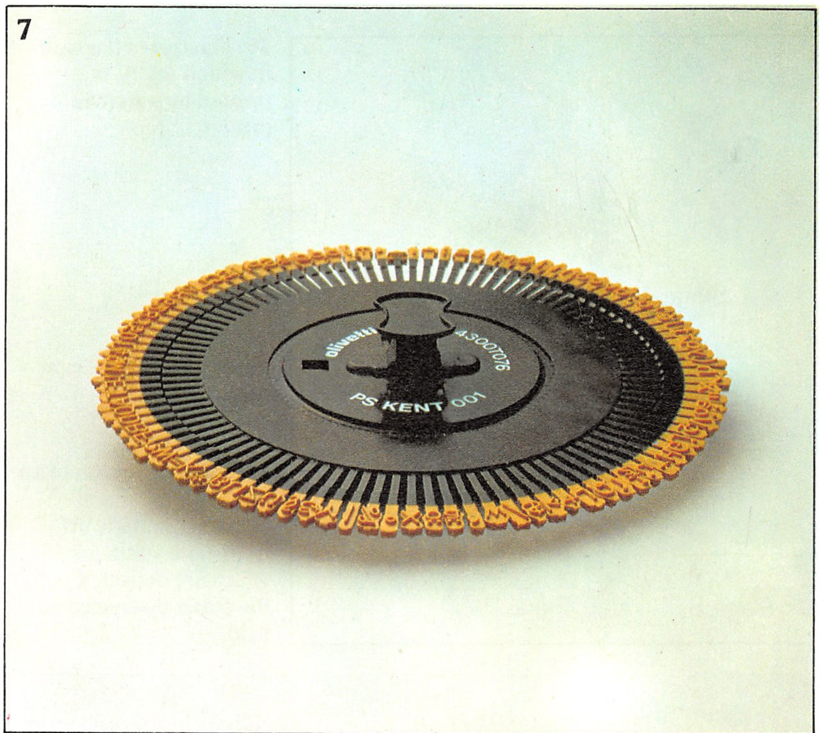
8. The daisy wheel rotates, the selected character is then impacted onto the paper by a hammer through an inked ribbon.

9. Detail of a matrix printer where a single column of dots moves across the paper to form the characters.

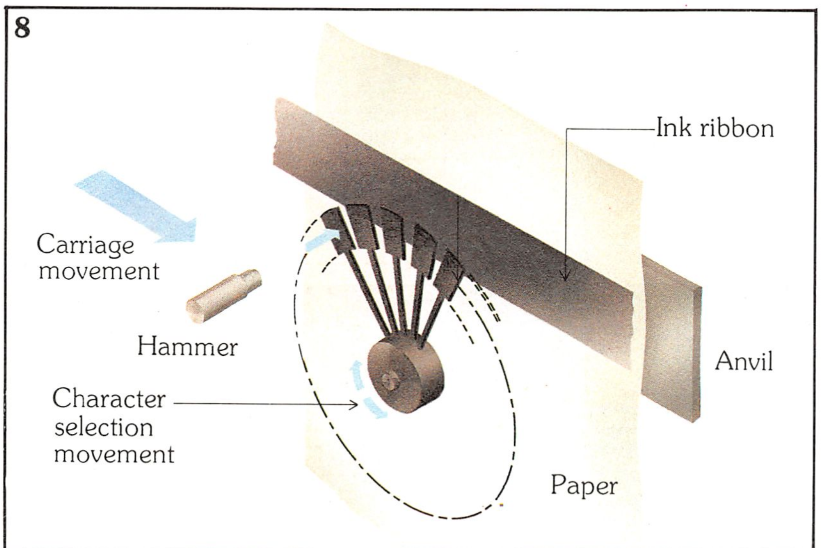


An example of a laser printer which is capable of printing both alphanumeric text and line drawings at a rate of 12 lines per minute.

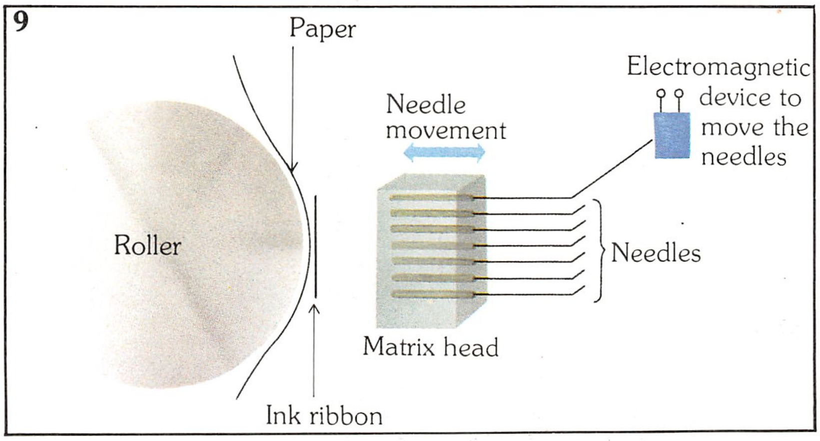
7



8



9



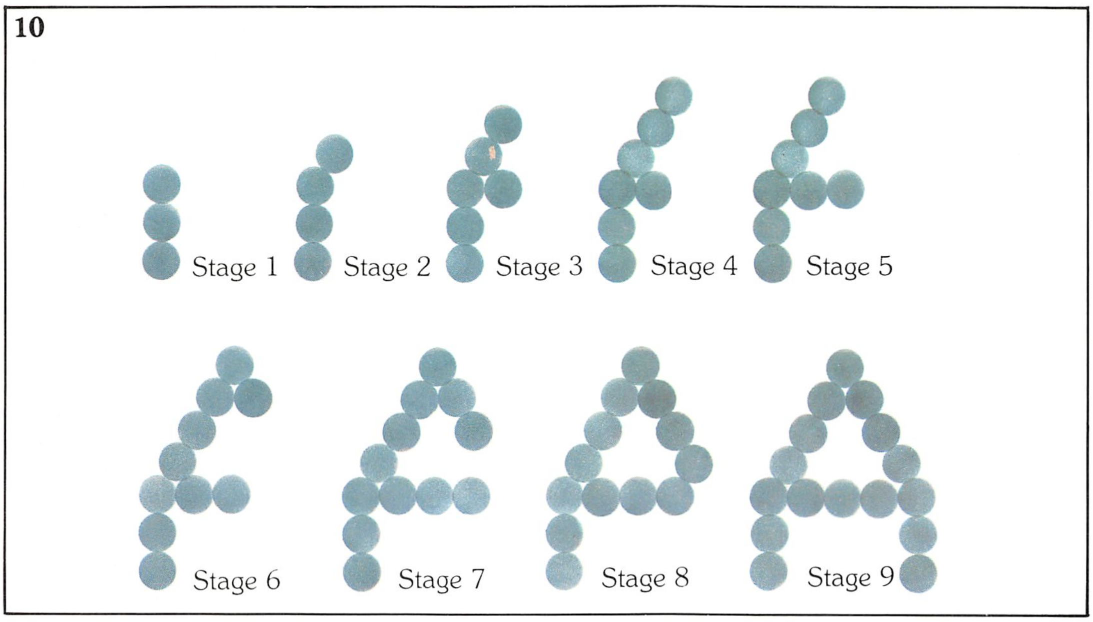
A variety of other types of impact printer are used in smaller personal or desk-top computers, or as part of the user terminals of large systems.

The **golf ball** type of printer has been extensively used for several years in both office typewriters and teletype printing computer terminals. Its basic operation is shown in *figure 6*. The print head rotates and moves vertically so that the required character is in the correct position for printing. An advantage of this type of printer is that the print head or golf ball can be changed easily, allowing a range of different typefaces to be used.

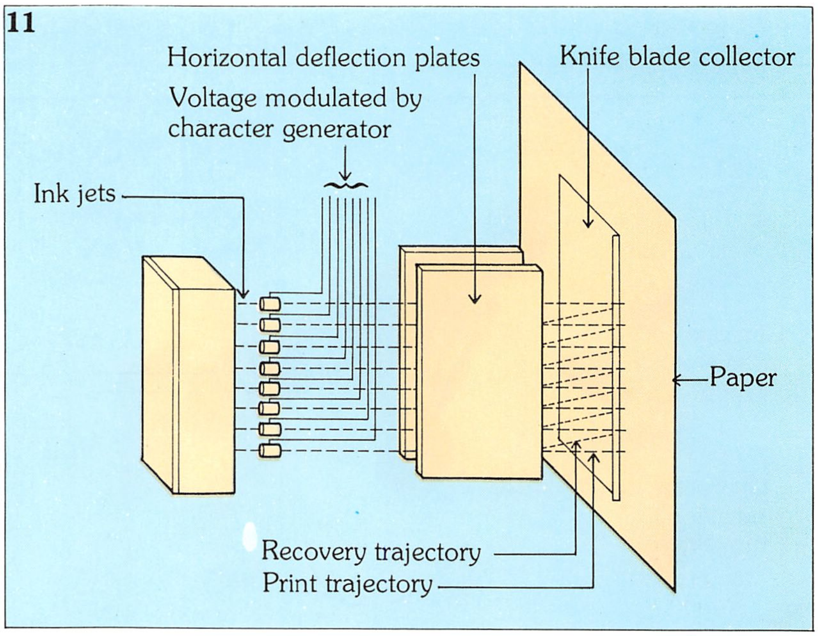
Another serial impact printer which is used in small systems – usually as an output device for word processors – is the **daisy wheel printer.** The type is carried on the tips of the spokes of a small metal or plastic wheel as shown in *figure 7. Figure 8* illustrates the nature of the printer's operation. Daisy wheel printers have the advantages of lightness, interchangeability of typeface, bidirectional operation and a very high quality of print (speed is up to 45 characters a second).

There are three types of **matrix printers,** all of which work on the same basic principles. If you look in detail at a piece of hard copy produced by a matrix printer you can see that the characters are made up of tiny dots. This is because they are formed by a matrix of needles, which are electromechanically controlled and pushed into the shapes of characters. These shapes press against the paper, transferring ink from a ribbon or carbon paper and thus printing the character.

The most basic type of matrix printer uses a single column of dots which moves across the paper forming the characters, this type is shown in *figure 9. Figure 10* shows how an 'A' is printed by this method. The more common matrix printers usually have a matrix of needles (7 × 5, 9 × 7, 7 × 7, 8 × 8), which can form and print one character at a time. Both these types are serial printers and can usually work as bidirectional machines. More sophisticated matrix printers have a matrix of needles long enough to print a line at a time without having to actually move across the paper. In other words it is a parallel printer. These machines can print



10. Illustrates the way in which an 'A' is printed by a matrix printer.



11. Mode of action of an ink jet printer. Each character is made up from eight jets of ink which are deflected onto the paper by an electric field.

at a rate of up to 350 characters a second.

Although matrix printers have been known to be fairly 'cheap and cheerful' machines it is now possible to get a much higher quality of print from these devices as more sophisticated matrix arrays are developed.

Non-impact printers

Examples of non-impact printers used as output peripherals include ink-jet, laser and thermal printers.

As the name suggests, an **ink jet printer** works by actually squirting jets of ink at the paper. The direction of these jets is controlled electrically by statically charging the ink jet and then deflecting it with an electrical field – in much the same manner as the cathode ray in a television is scanned across the screen. As you can see from *figure 11*, there are in this case eight jets of ink which make up the full height of a character. These printers can operate quickly – up to 250 characters a second – and are highly accurate.

More sophisticated types of ink jet printers eject only one drop of ink at a time. The output looks like that of matrix printers, and they can be programmed to print in any colour.

Laser printing technology is based on a continuously rotating, selenium covered drum. Selenium is a substance that acts as an insulator if it is kept in the dark and as a conductor when exposed to light. A laser beam is directed onto the drum in the inverse shape of the character to be printed. One line of characters is scanned at a time. All the parts of the drum that have been scanned by the laser are made conductive and are electrostatically charged. These charged areas attract and retain ink dust, toner, which is then transferred to the paper as the drum rotates. A system of compression and heating ensures that the toner remains bonded to the paper. This is a parallel printing method and speeds up to 13,000 lines per

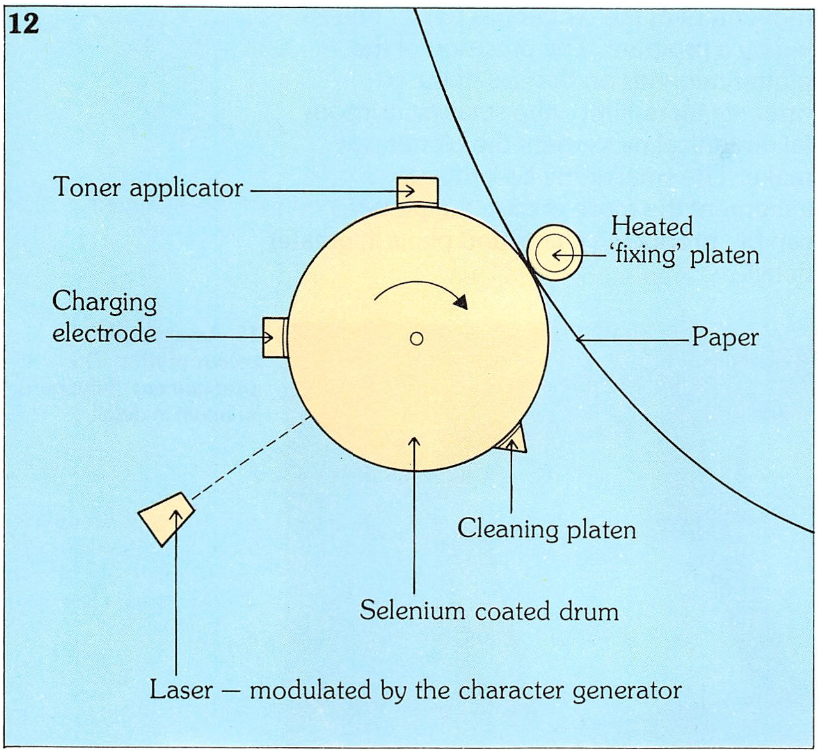
minute can be achieved. The basic layout of a laser printer is shown in *figure 12*.

Thermal printers are low-cost non-impact machines. They use a matrix of heating elements to print characters on special paper. *Figure 13* shows the arrangement of this print head. While these devices are quite cheap, they are slow and do not give a very high quality of print. They are usually used in small home computing systems.

In general the data to be printed by any device is transferred from main memory one line at a time and coded so that it can be accepted by the peripheral. Once coded the data is held in a print buffer, from which it is read by the printer, usually a line at a time.

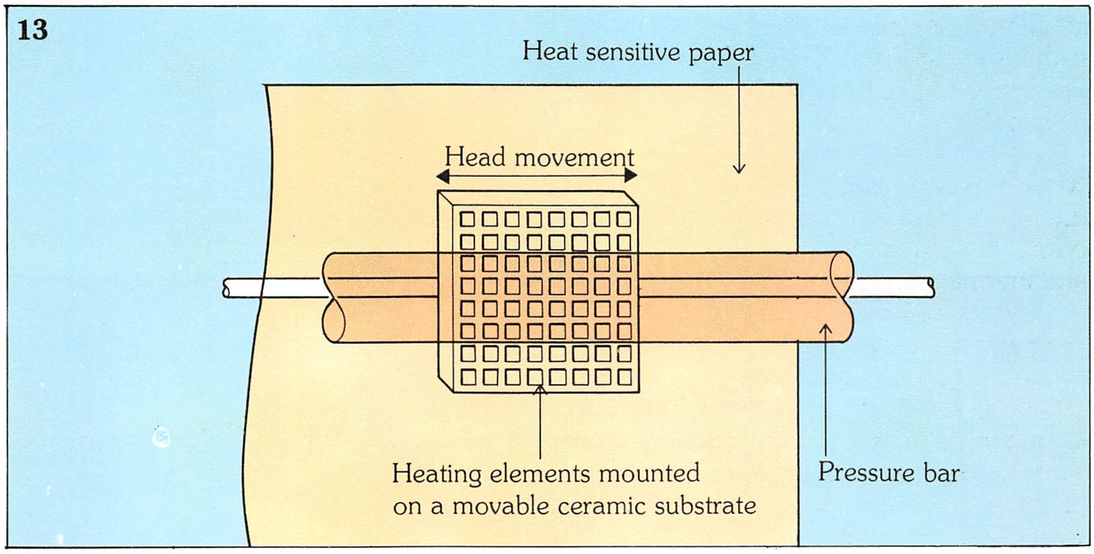
If the machine is a matrix printer then special graphics can be printed as well as ordinary characters. In this case the print buffer would take the form of a matrix. In the case of the 8 × 8 matrix, there would be 8 bytes in memory which would represent the character to be printed. *Figure 14* shows how the omega symbol is made up as binary code and as a matrix print out. Binary 1 indicates print. In fact any combination of dots within the matrix can be programmed, enabling users to design and use their own special characters.

12



12. Basic design of a laser printer showing the selenium coated drum.

13



13. The printing head of a thermal printer.

14

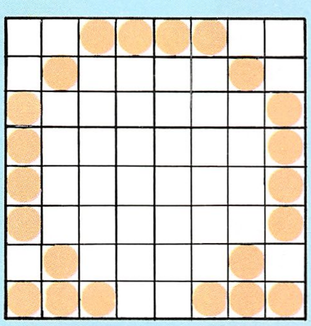


Image of the Ω character

0	0	1	1	1	1	0	0
0	1	0	0	0	0	1	0
1	0	0	0	0	0	0	1
1	0	0	0	0	0	0	1
1	0	0	0	0	0	0	1
1	0	0	0	0	0	0	1
0	1	0	0	0	0	1	0
1	1	1	0	0	1	1	1

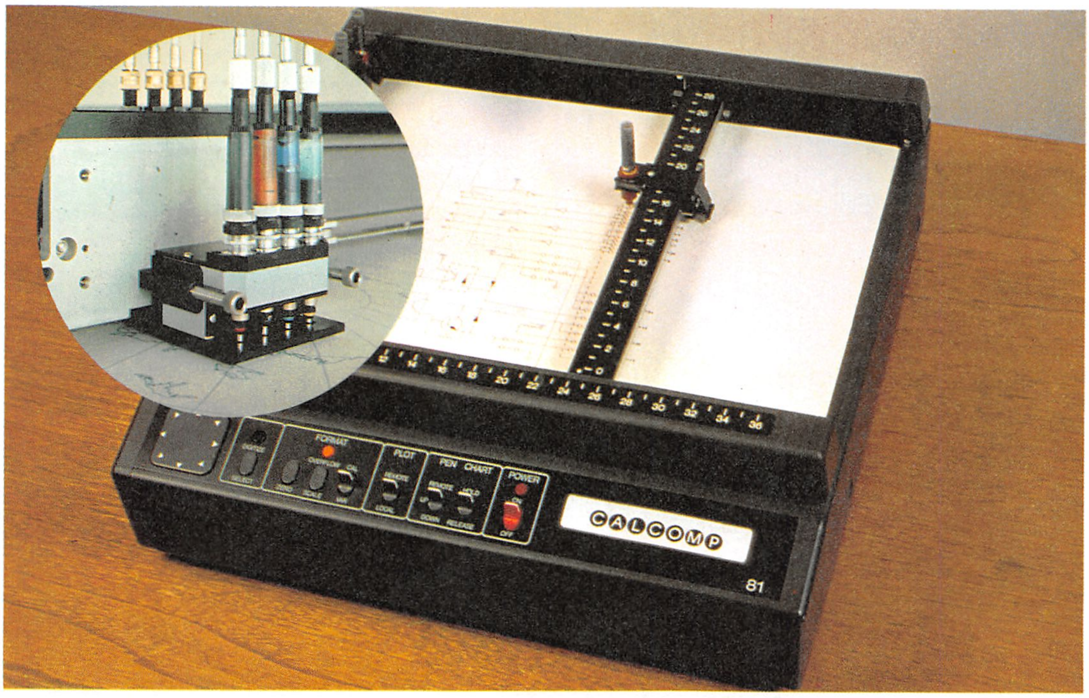
Contents of the corresponding bytes

14. The omega character as a matrix printout (left) and as binary code (right).

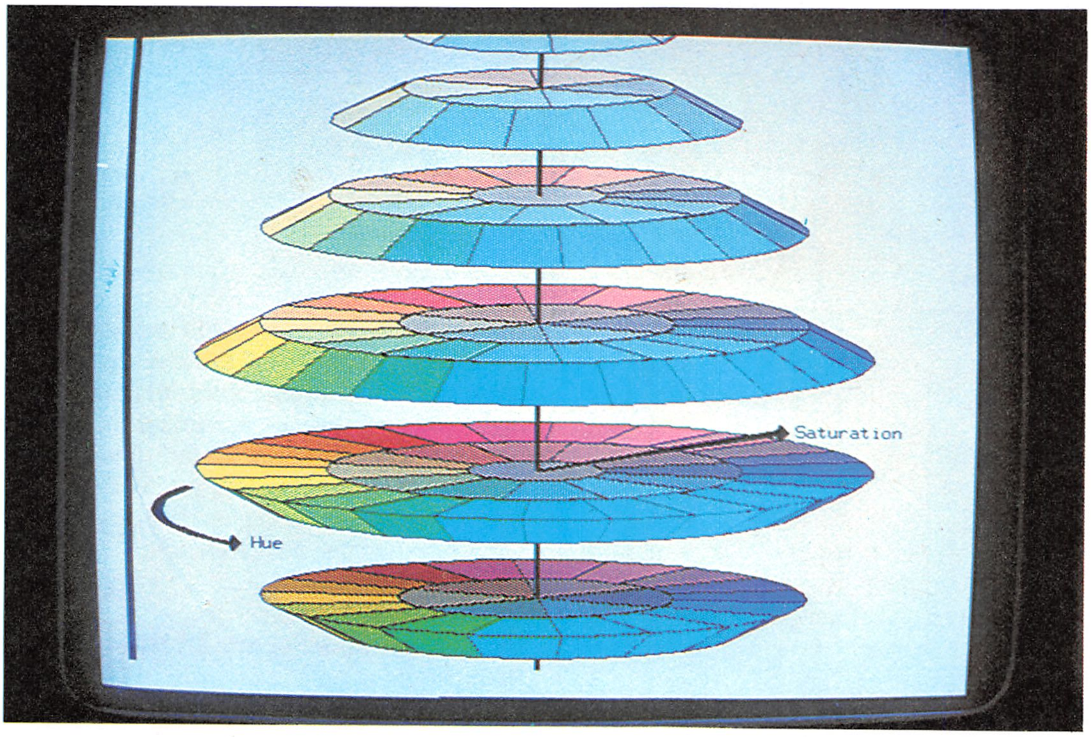
Plotters

Plotters are more like drawing machines than printers and can be used to produce high quality graphic output, such as graphs

and plans. *Figure 15* shows a small plotter. As pens are used, changing ink colour is achieved by changing pens. Most plotters work by moving the pen along the horizontal and vertical axes of the paper. Other simpler plotters just move the pen vertically up and down, the horizontal movement being carried out by the paper itself which is moved on rollers. In either case the movement of the plotter has to be controlled by a program. The precision of the plotter depends on the size of the co-ordinate increment – the smallest horizontal or vertical movement the plotter can make. The smaller the co-ordinate increment the more precise the plotter can be, producing charts and plans in greater detail.



15. An example of a colour plotter. The arrangement of the pens is shown in detail.



16. Colour output from a graphics demonstration program.

The visual display unit

The visual display unit, or VDU, is basically a television screen that can show computer output. The screen is usually linked to a keyboard so that it can be used simply as an information retrieval system or as a user programming terminal.

In either case the screen can be monochrome or colour depending on the application it is used for. Colour screens are used mainly in applications demanding a high graphic capability – such as engineering design or computer games, while monochrome is adequate for most office work. *Figure 16* shows the output from part of a graphics demonstration program.

The quality of the information and the number of characters that can be displayed depends on the resolution of the screen. The screen area is made up of many tiny dots, called **pixels.** The pixels make up a matrix and the characters and graphics are created by electronically illuminating them. The more pixels there are on the screen, the higher the quality of the display. If you think back to the matrix printer you can see that an 8 × 8 matrix can produce more intricate characters than a 7 × 5 matrix.

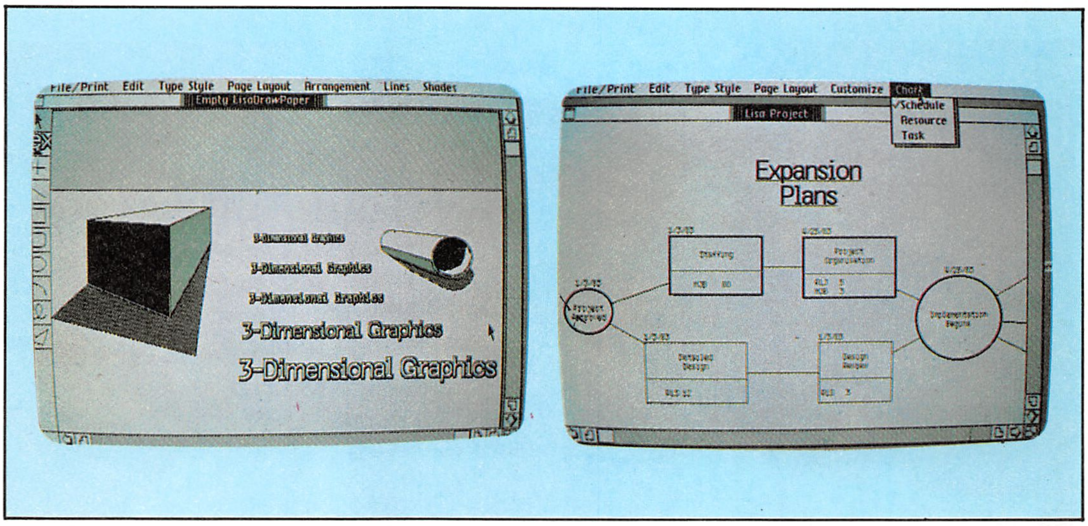
Screens which use a lot of pixels are called **high resolution** devices. In the same way as each needle in a matrix printer has to have a bit of memory allocated to it, each pixel on a screen needs a memory position – more than one if colour is used. As with matrix printers, users can design their own characters by defining pixels into a shape.

The number of characters that can be displayed at a time thus depends on the computer's allocated memory capacity and not the actual screen unit. Normal VDUs can have up to 80 characters per line and around 24 lines. If smaller characters are used, then more lines can be displayed.

As characters are displayed on the screen, the writing will 'scroll' – as more is added to the bottom of the screen, the top line disappears, making room for a new line on the bottom. This text is not lost – merely invisible. Most terminals can scroll both ways.

When a keyboard is used with a VDU, a flashing dot or square called a **cursor** moves along the screen as information is fed in. This indicates the position of the next character, and can usually be moved to any point on the screen, e.g. to make correction to the displayed text.

The VDU itself can also act as a direct input device when used with a **light-pen.** This device is light sensitive and can be used to 'pick out' points on the screen. It is connected so that the computer can identify which part of the screen it is placed against. The light-pen can be used simply to select items from displayed lists so as to obtain more detailed information, or to actually draw onto the screen. When employed as a drawing device the light-pen has many useful applications in designing integrated circuits, for example, as well as engineering and architecture.



Right: examples of new screen techniques illustrating the use of graphics and the 3-D approach. (Photo: Apple Computer).

Terminals

A terminal is basically a user operated input and output device, used to communicate directly with the computer. Terminals are made up of a typewriter style keyboard (*figure 17*) and either a VDU or a printer.

We've already explained the use of VDUs and printers, so we'll move on to look at how a keyboard operates as an input device.

Each key on a keyboard is a switch. If you imagine the keys as a matrix, pressing a key closes the contact at the intersection of a row and a column. The rows of the matrix are output lines and the columns input lines. The device's control unit sends a voltage (logic 1) along the rows of the matrix. It does this cyclically, a row at a time, thousands of times a second. When a key is pressed this voltage is sent along the corresponding column, coded – usually in ASCII code – to be received by the CPU as input. By systematically scanning the columns the key that has been pressed can be identified.

If several keys are pressed at once, this gives an invalid output which is ignored. As the keyboard is continuously scanned, the next valid output is accepted.

A problem associated with keyboards is what is known as 'keybounce'. When a key is pressed it takes a little time to be effectively and fully closed. In fact there is a delay of about 15-20 ms during which a number of oscillations occur. The same thing happens when the key is released. These oscillations, if not effectively dealt with, could be interpreted by the computer as legitimate keystrokes, causing errors in the input data. It is therefore important to ensure that the keystroke is not recorded in the intermediate state. Filters or delay routines can be used to remove the effects of keybounce.

The devices we have dealt with so far have been non-intelligent terminals which deal exclusively with input and output. There are also terminals known as intelligent or **smart terminals** which can be programmed independently from the larger system of which they are a part. These intelligent terminals are, in fact, small computers which can function autonomously.

17. A terminal comprises a typewriter style keyboard and either a VDU or a printer. This one illustrates the new technique of multiple windows on the display into separate computer processes. (Photo: Apollo Computer).

Remote Data Transmission

Terminals can be connected to a computer system in two ways. If they are sufficiently close to the main computer they can be directly linked by cables which are laid by the owners of the computer system. These are high speed lines. These are known as **non-remote terminals.**

Remote terminals, on the other hand, are used when a direct link is impractical because the distance between terminal and computer is too great. In this case data transmission can take place by telephone lines (special data transmission lines can be rented from British Telecom in the U.K.) or by radio link. This system is very useful for businesses which have offices in several locations, all of which needing access to one central computer.

Data is transmitted serially to and from remote terminals, that is to say the bits are sent one after another. The transmission can either be synchronous or asynchronous depending on the systems and type of link used. Radio data links usually give medium or high speed transmission times, which make them suitable for synchronous transmission. Telephone lines on the other hand provide low speed links and asynchronous transmission is used. **Synchronous transmission** is controlled by a series of clock pulses which ensure that the data bits are sent and received at regular time intervals. The data transmitted is arranged in blocks, at the end of which is an **end of block character**, which indicates the end of that part of the transmission and enables the receiver to respond.

Asynchronous transmission is not synchronised to a clock pulse, but takes place in its own time. Information traffic can only move in one direction at a time, so at each end of the data a number of bits signify the beginning and end of transmission; this indicates that the receiver can then reply.

Synchronous transmission is generally used in medium and high speed lines, while asynchronous transmission uses low speed links (like telephone lines). The speed of transmission is measured in bits per second (this unit of transmission speed is often referred to as a baud, however this is not strictly correct).

Other peripherals

Many other types of peripheral device can be used in computer systems. They usually have very specialised applications and are not part of general data processing operations.

Character and mark readers

These peripherals are mainly used to cut down the labour costs incurred by transferring data from documents to tape or disk by hand.

Character readers can, as their name suggests, read printed or handwritten characters. There are two basic types – **magnetic ink character readers** (MICR) and **optical character readers** (OCR). MICRs are used to recognise special characters printed in magnetic ink – the main application being the serial numbers found on the bottom of cheques. Magnetic ink is used because it can be written over or creased without affecting its legibility to a reading machine.

Optical character readers are used to recognise characters which can be either printed in a special typeface, typewritten or handwritten, depending on the sophistication of the machine used. These systems can be useful for instance in checking standard business documents or analysing the answers given to research polls.

Mark readers are widely used in checking responses on the type of multiple choice forms found on exam papers or research polls. Standard forms are used with boxes corresponding to the various answers or responses. The boxes are usually filled in with a soft pencil; this can be recognised by the machine either optically or electrically, as graphite (the main constituent of pencil 'lead') is electrically conductive.

Joysticks

Joysticks are familiar to most people, as they are used in connection with computer games. The stick can be moved in any direction to control a point (cursor) on a screen. But they're also found in industrial or business applications, where the cursor needs to be moved around the screen, say to select items from lists or to amend drawings.

Mice
'Mice' can be used in much the same way as joysticks, to move the cursor around a screen to select items from displayed lists or to draw on the display. A **mouse** consists of a small casing with a rolling ball set under it, and a switch or switches on the top side. Moving the mouse around a desk-top makes the ball rotate, rather like how the ball in a ball pen moves when writing. The movement of the ball is translated into variations of the horizontal and vertical co-ordinates of the cursor.

Graphics tables
The graphics table or digital table (*figure 18*) is a device used in engineering applications. Anything that is drawn on a piece of paper placed on the graphics table is transferred into digital information which a computer can store and manipulate. One great advantage of computer graphic systems is that drawings can be electronically transmitted around the world. This is easily done because drawings are stored in computers in the same way as ordinary data – as strings of binary digits. Plans and diagrams can be transmitted as easily as written information and this method is both quicker and safer than postal services.

Speaking computers
Considerable development work has been put into making computers speak. The applications of such systems are very wide, ranging from aiding the handicapped to providing essential information for motorists from the 'talking dashboard'.

Whereas aural computer output is now relatively easy to achieve, the use of speech as input is rather more difficult, as such systems have to contend with the very different ways in which people speak. Some machines do exist, although at the moment the vocabulary range which they can accept as input is very limited and has to be defined by the user. However, once set up in this manner they can work very effectively.

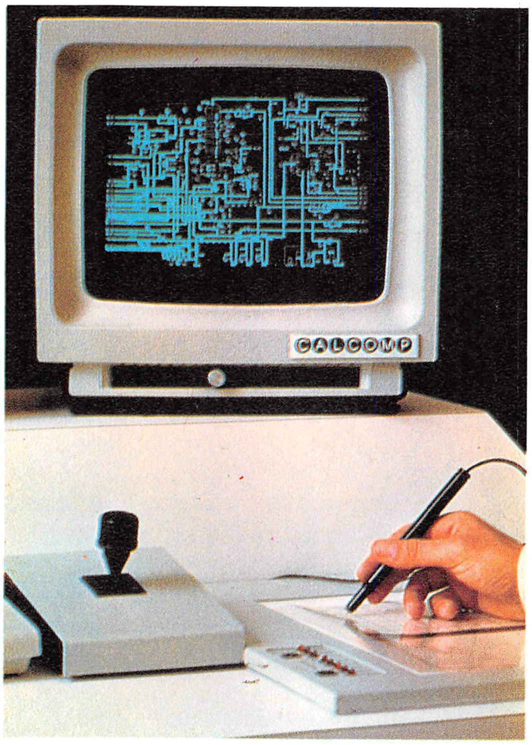
Secure systems is one area where voice input is especially useful. Voice patterns for each individual are different – the computer holds a sample pattern in memory and then matches it against that of the speaker.

Input/output techniques

Having looked at the common input/output devices let's now examine some of the main techniques that are used for the transfer of data to and from the peripherals of a modern microprocessor system. A peripheral is usually connected to a computer via an **interface**. This transfers the data in series or parallel, and deals with any necessary conversions between series and parallel data transmission and the various data codes used.

Most peripherals have complex mechanical parts and therefore contain a controller, which as its name suggests carries out all the control functions necessary for the correct operation of the device. For example, the controller of a floppy disk unit must at least perform the following functions:

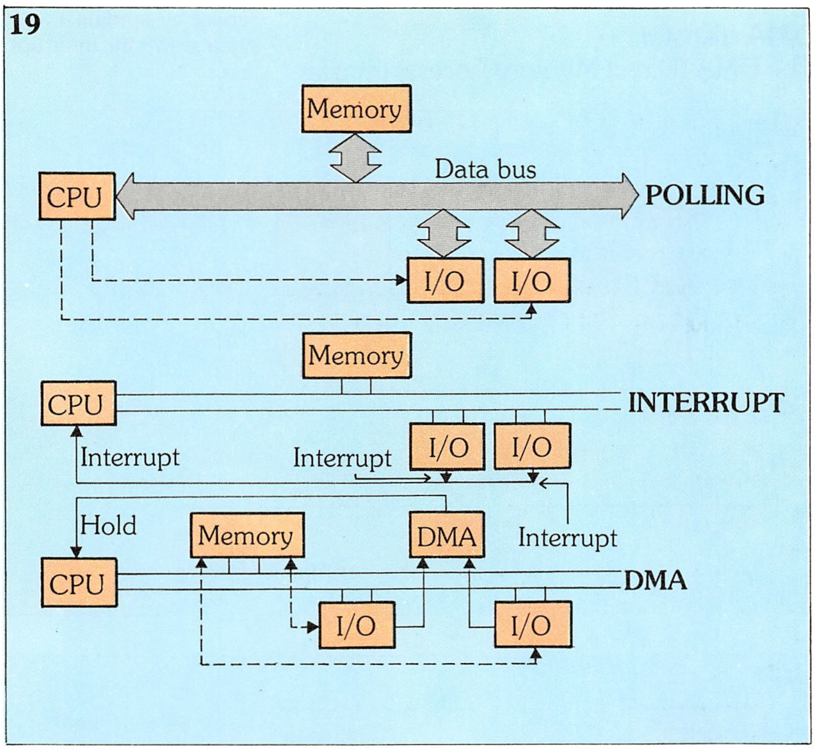
Reading
1) Track register loading: find out which track the required data is stored on;
2) Search generation: search for this track;
3) Wait for correct positioning;
4) Transfer of data under interrupt of control;
5) Check that the operation is being carried out properly.

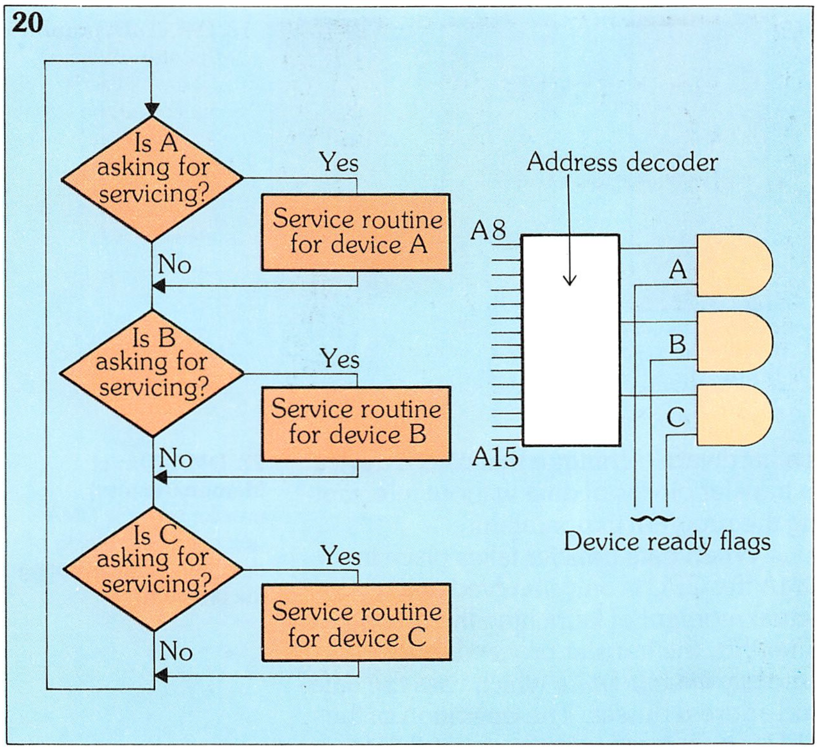


18. Detail of a graphics workstation showing how a pen is used to input information through the graphics table.

19. Examples of three dialogue procedures: polling, interrupt and DMA.

20. The set of routines comprising the polling sequence.





Writing
1) Track register loading: find out which track the data is to be stored on;
2) Search generation: move to this track;
3) Wait for correct positioning;
4) Data is written onto the disk when the unit receives the request signal;
5) Verification of status indicators for BUSY and check character (CRC) block.

The controller for a video screen must handle the process of searching in the character table for the dot (pixel) matrix needed to represent characters and control the cursor and scrolling.

Most computer systems will have several peripherals in use at any one time. To allow these to work, a dialogue procedure or I/O technique must be established for the system. Three dialogue procedures: polling (programmed I/O), interrupt and DMA (direct memory access) are shown in schematic form in *figure 19*.

Polling

Programmed I/O or polling is the simplest I/O technique. It is a synchronous (controlled by a clock pulse) technique in which the computer cyclically polls (checks) the data bus, the address bus and some lines of the control bus to see if they need servicing. (When we talk about servicing we mean arranging the transfer of I/O data.)

The connecting interface of each device has a flag register which is examined to see whether or not the device is waiting to be serviced. *Figure 20* shows a schematic diagram of the polling sequence, which is built into the software of the CPU's operating system.

The question and answer procedure between computer and peripheral is known as **handshaking** or **protocol**. This is a type of communication between two devices in which the transfer of data is accompanied by control signals which effectively communicate a request for data from one of the devices, the data availability on the part of the other and finally the reception of the data by the first one. Handshaking is a two-way dialogue and is used for both the input and output of data to and from the computer.

If a peripheral tells the computer that it is ready to send data, the computer must check that the contents of the peripheral register have been updated before reading the data. This is to stop the computer reading the same data twice. Likewise, if the computer sends data to the peripheral, it must first verify that it is ready to receive it, send it, and then check that the data has

been received.

The technique of polling has both advantages and disadvantages. The polling cycle can be slow – the more machines there are to be polled the longer it will take to complete each cycle. The time taken to service each machine will also add to the delay. If any peripheral requests service immediately after it has been polled, it must wait for the whole cycle to be repeated.

On the other hand, polling is a software controlled communication technique and requires almost no additional hardware.

Interrupt procedure

The interrupt procedure is one that enables a peripheral to literally interrupt the program in execution when it requires service. *Figure 21* shows how the peripheral requests service by sending a signal down the 'interrupt line'. The computer recognises the request (2 in the diagram). It suspends the program being executed, serves the peripheral (3 in the diagram) and then returns to the program which was interrupted.

When the CPU receives an interrupt, it has to preserve the state of the main program in progress. This is done by storing – in a special memory zone – the data necessary to continue the program once the interrupt is completed. It also stores the contents of the PC (program counter) so that the address of the next instruction to carry out is known, and the contents of the status register.

After this operation, known as **back-up**, the peripheral which has requested the interrupt is serviced with a suitable routine. When this is completed, the control returns to the original program, resets the conditions (PC, status register) and picks up precisely where it left off.

Interrupt is an asynchronous control, that is to say it is not controlled by the program. This type of I/O control is more complex than polling because the CPU can't predict when data transfer is going to take place. Another factor that has to be taken into account is **service priority**.

If a system uses more than one peripheral, then each device has to be assigned a level of service priority, to ensure that if two or more devices request service at the same time they are dealt with in order of importance. This is known as **queuing**.

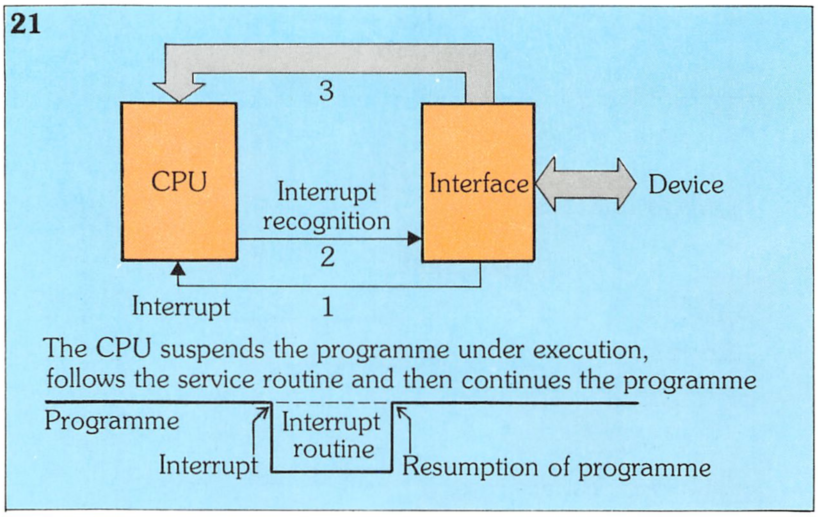
This I/O technique needs more complex hardware and programming than polling, but it can be extremely fast.

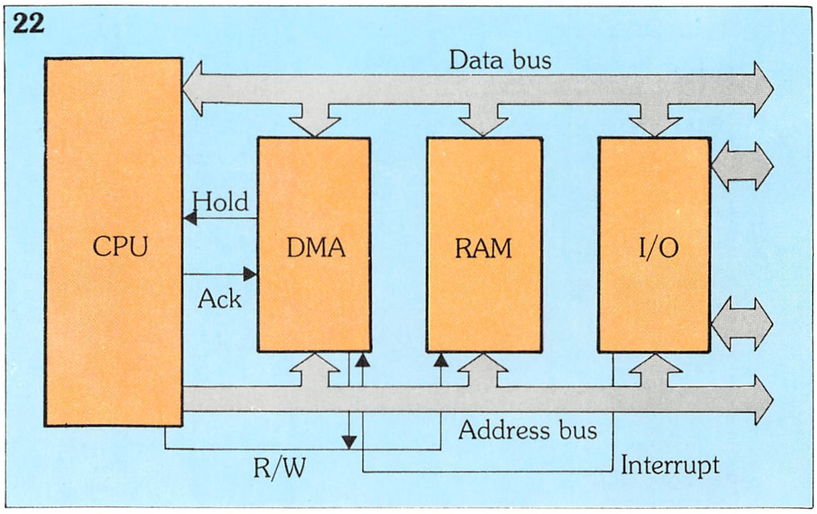
DMA transfer

The DMA (Direct Memory Access) transfer is a hardware technique that uses a device to transfer blocks of data without interrupting the program in operation.

When data transfer takes place in DMA the CPU is only involved at the actual moment of launching the operation. After this, the transfer proceeds autonomously via hardware which uses the data and address busses. The operation of the CPU is only momentarily suspended to

21. The interrupt procedure. The peripheral requests service by sending a signal down the interrupt line.





22. DMA (Direct Memory Access) transfer uses the DMA unit to transfer blocks of data without interrupting the program.

free these busses for the DMA transfer. The signal that suspends the CPU operations is known as **hold**. The answering signal is known as **ack** (acknowledgement). While DMA is strictly an interrupt technique, it does not depend on the central processor to control its operation. The operation of direct memory access depends on a particular device – known as the DMA unit. *Figure 22* summarizes this procedure in a diagram.

As with ordinary interrupt transfer, a system of priorities and queuing is needed for DMA to manage the allocation of access to each peripheral. The CPU is informed by an interrupt when data transfer has ended. DMA units are expensive devices and are normally used when very rapid data transfers are needed – usually for disks, tapes or VDUs.

The programming needed to carry out I/O procedures in a computer is very complex, so manufacturers always supply standard programs with the computer, to control the peripherals. This set of programs is called the **I/O control system.**

Glossary

card reader	a machine for reading the data represented by the holes in punched cards
cursor	pulsating indicator which appears on a VDU screen
DMA	direct memory access, a form of dialogue procedure between computer and peripheral
interface	a general term used to describe the linking device between any two units – often the CPU and a peripheral
interrupt	a break in a program caused by a peripheral requiring that control should pass temporarily to another routine
I/O buffer	a memory which temporarily stores data which is being transferred from the CPU to peripherals
IOCS	input/output control system. The set of programs supplied with a computer which controls the peripherals
I/O technique	method of communication between the CPU and a peripheral
joystick	lever which can be moved up and down and to the right and left, enabling the user to move the cursor quickly around the screen
parallel printer	printer which prints, or appears to print, all the contents of a line simultaneously
plotter	kind of printer particularly useful in graphics applications
polling	the term used to describe the action of the CPU when it addresses each peripheral in turn to see if it requires servicing
serial printer	printer which prints character by character along a line
status register	set of bits conveying the current status of a particular peripheral
terminal	any point at which data may be input to or output from the computer
VDU	visual display unit. The video screen which is used in most terminals

BASIC THEORY REFRESHER

ELECTRICAL TECHNOLOGY

Flux, force and energy

In the Basic Theory article on electric field we saw that when a voltage is applied to two objects – say the two plates of a capacitor – an electric field of magnitude K (measured in volts per metre) is induced in the space between them. The magnitude of this electric field is determined by the intensity of the applied voltage and the distance between the two plates.

The voltage applied to the plates of a capacitor is directly proportional to the charge which is induced on them. The value of this induced charge is dependent on the permittivity of the insulating material (the **dielectric**) filling the space between the two plates; the greater the permittivity, the larger the induced charge.

How is this charge set up and how does it depend on the permittivity of the dielectric material. In order to explain this we need to consider the **electric flux** (meaning the flow) from the positive to the negative plate of the capacitor. This flux is set up by the electric field (K) created by the potential difference between the plates, and its magnitude is exactly equal to the charge (Q-measured in coulombs), induced on the plates of the capacitor. We can therefore say that:

$$\Psi = Q$$

where Ψ is the total electrical flux.

Flux density (D) is defined as the total amount of flux flowing through an area of $1m^2$ placed perpendicularly to the electric flux, as shown in *figure 1*.

If the cross-sectional area of the capacitor plates in *figure 1* is equal to Am^2, and they carry a charge of Q coulombs (which is equal to the total flux (Ψ) between the plates) then:

$$\text{flux density } D = \Psi / A$$

and is measured in Cm^{-2}.

Permittivity

Since the electric flux (Ψ) is dependent on the field strength K and the material of the dielectric, we may show that:

$$D = \varepsilon K$$

where ε is the permittivity of the dielectric. We have previously seen that permittivity is measured in Farads per metre.

Coulomb's law

In the basic theory article on electrical fields, we saw that like charges repel and opposite charges attract each other. In either case the two bodies exert a force on each other, and this force can be expressed mathematically. If we place two very small bodies with a charge of $+q_1$ and $+q_2$ coulombs respectively at a distance of d metres apart, each body will be repelled from the other by a force (F). This can be written as:

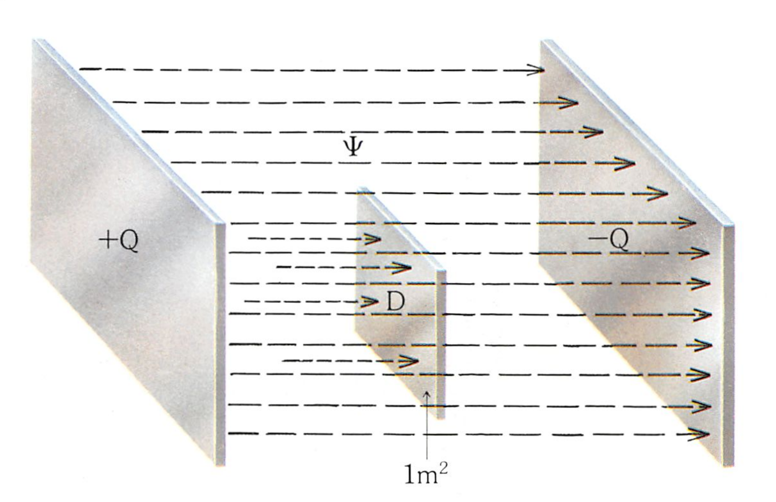
$$F = \frac{q_1 \times q_2}{d^2}$$

The force F will be measured in **newtons** (N). (One newton is the force required to accelerate a mass of 1 kilogram by 1 metre per second per second.)

The formula for the force between two charged bodies is called an **inverse square law,** as the force is inversely proportional to the square of the distance apart. This type of law applies to other physical situations in nature, for example the force of gravity.

1. The flux density is the total amount of flux flowing through an area $1\ m^2$ placed perpendicularly to the electric flux.

1



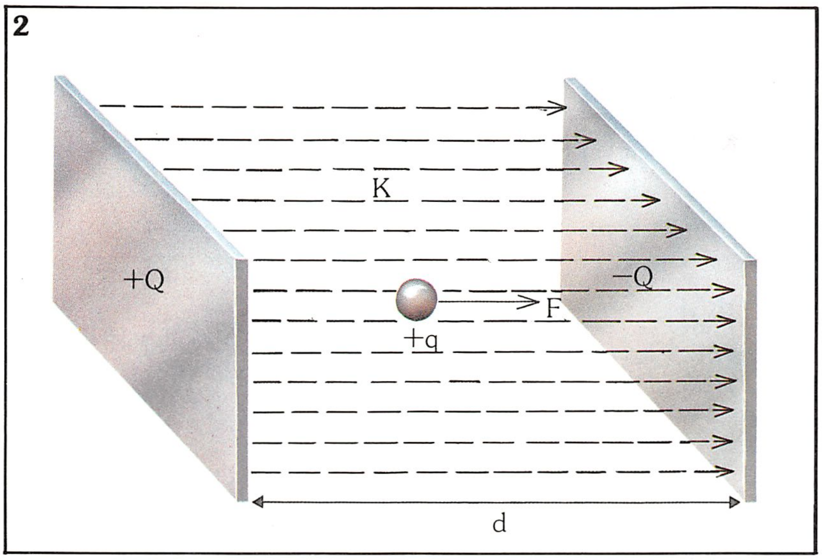
Force on a charged body in an electric field

As a result of this force between two charged bodies, a particle of charge +q placed between the plates of a capacitor will be repelled by the charge on the positive plate and attracted to the negative plate. This situation is shown in *figure 2*. The force (F measured in newtons) on the charged body may be expressed in terms of the electric field (K volts per metre) between the plates:

$$F = K_q$$

Because the plates of a capacitor are oppositely charged, they will naturally attract each other in the same manner.

Every time you switch on a television set you can see a practical example of this phenomenon; electrons – small negatively charged particles – are accelerated along the picture tube to the screen, where they cause



2. A particle of charge, +q, placed between the plates of a capacitor.

the screen's phosphor coating to glow. The movement and direction of these electrons is caused by an electric field that is set up in the tube.

Energy stored in a capacitor

As an electrically charged body placed between the plates of a capacitor is acted on by a force, we can see that this force accelerates the body and so does work on it. Because the energy involved cannot leave the capacitor until it is discharged, we can see how the capacitor is capable of storing electrical energy. The amount of this energy, E, measured in joules is given by:

$$E = \tfrac{1}{2}C \times V^2$$

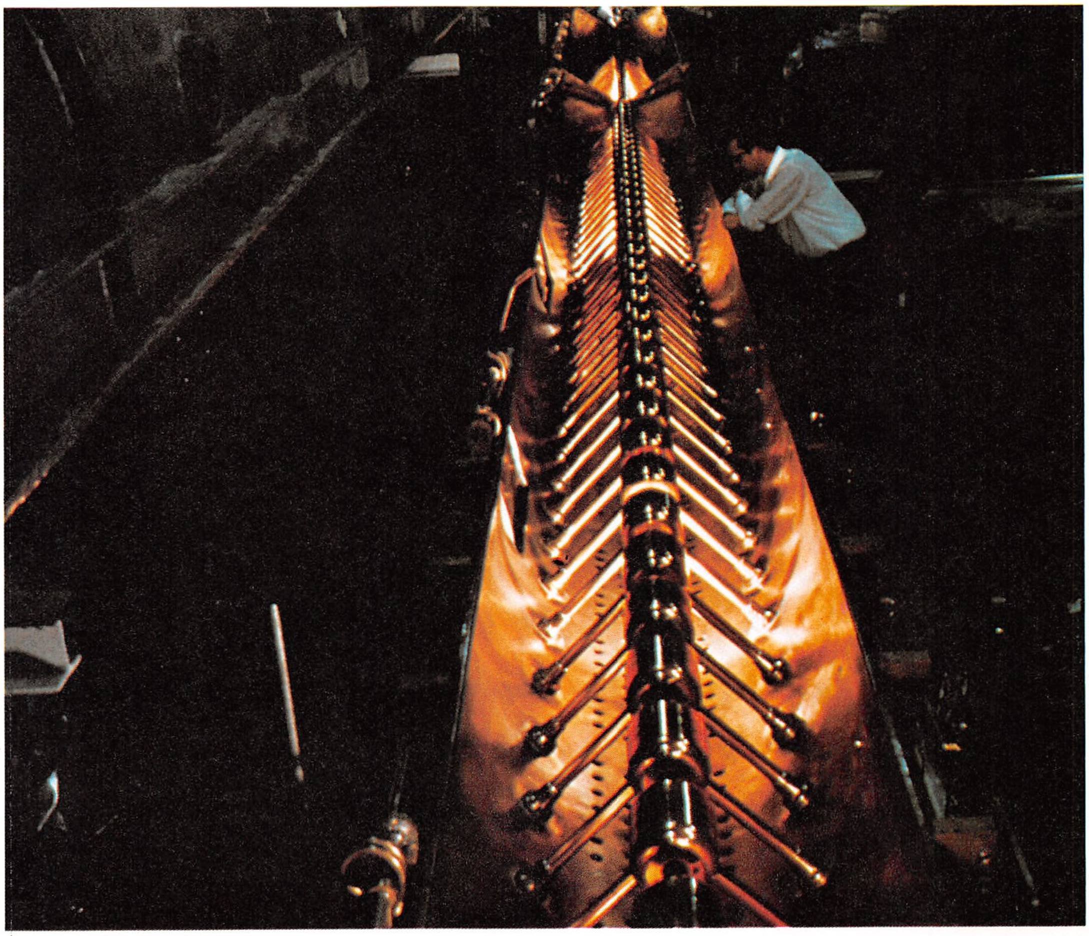
where C is the capacitance (measured in farads), and V is the voltage across the capacitor.

The ability of a capacitor to store energy demonstrates a fact that is fundamental to the majority of electrical and electronic circuits. The actual energy is stored in the electromagnetic field. In the case of a capacitor, this field exists between its plates, but it is not necessary to use a capacitor to generate an electromagnetic field. There is energy in the field of a radio wave, even though there are no physical circuit elements anywhere nearby.

The energy stored in a cubic metre of an electromagnetic field can be found by using the equation:

$$E = \tfrac{1}{2}K \times D$$

where K is the electric field strength in Vm^{-1} and D is the electric flux density in Cm^{-2}. □



Particle accelerators use the principle that an electric field accelerates or deflects positive or negative particles parallel with the direction of the field. This one is at Cern, Switzerland.

Science Photo Library

BASIC THEORY REFRESHER

ELECTRICAL TECHNOLOGY

Magnetic fields and flux

We know from the Basic Theory article on magnetism that a current flowing through a circuit sets up a magnetic field around that circuit. How is this field related to the current that causes it? Because the electric current and the magnetic field are related, we should expect that an increase in the current will increase the magnetic field proportionally. We should also expect the magnetic field to diminish as it gets farther away from the wire carrying the current.

If we take the example of a long straight wire that has a current flowing in it, the electric field strength H, measured in amperes per metre (Am^{-1}), at a distance r from the wire, is given by:

$$H = \frac{I}{2\pi r}$$

where the current I, measured in amperes, is divided by the circumference of the circle of the magnetic field around the wire. From this, we can see that the field strength H is directly proportional to the current I, and falls off as the distance r from the wire increases.

A coil is a common electromagnetic circuit and *figure 1* shows a single turn coil of radius r metres. Applying the corkscrew rule we see that every small section of the wire will give rise to a magnetic field. Two such sections with magnetic fields are shown in *figure 1*. The total magnetic field (H) at the centre of this single turn of wire is given by:

$$H = \frac{I}{2r}$$

A long coil of wire that has many turns is called a **solenoid** and solenoids are widely used as electromagnets in many applications. A cross-section of a solenoid is shown in *figure 2*. The magnetic field strength, H, at any point inside a solenoid is given by:

$$H = \frac{NI}{l}$$

where N is the number of turns of wire and l is the length of the coil in metres. An interesting point to note is that field strength is not dependent on the diameter of the solenoid. As there are a number of turns of wire in the coil the field strength, H, is measured in units of ampere-turns per metre (Atm^{-1}). A field strength of 1000 Atm^{-1} is defined as existing when a current of 10 A flows through a solenoid that has a length of 100 turns per metre.

Toriods

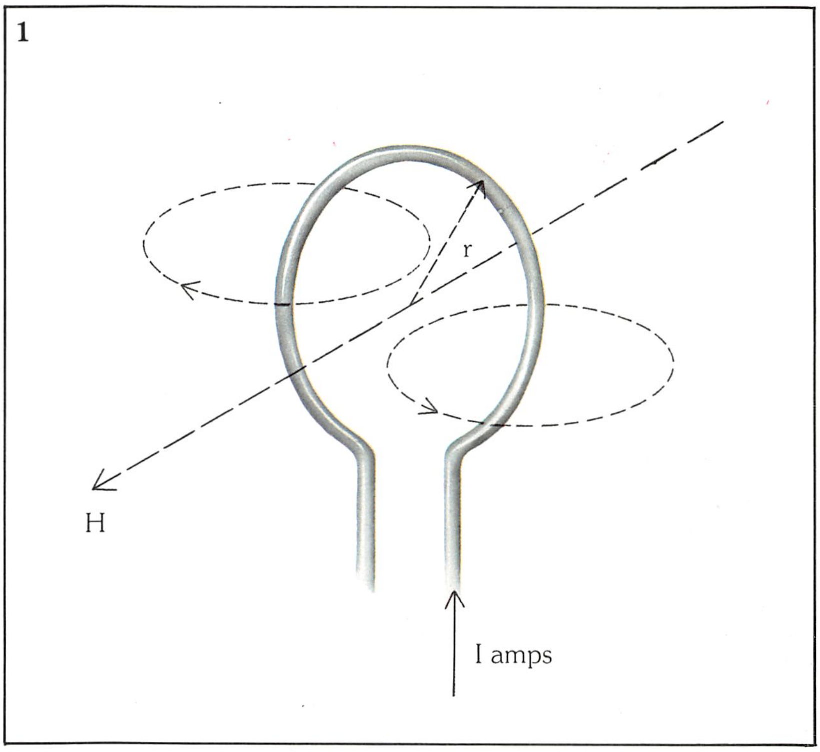
If the solenoid is bent round into the shape shown in *figure 3*, it is said to form a **toroid.** The field strength can be found by using the equation above, but the length of the coil is now given by $2\pi R$ – where R is the radius of the toroid measured in metres. This gives us the expression:

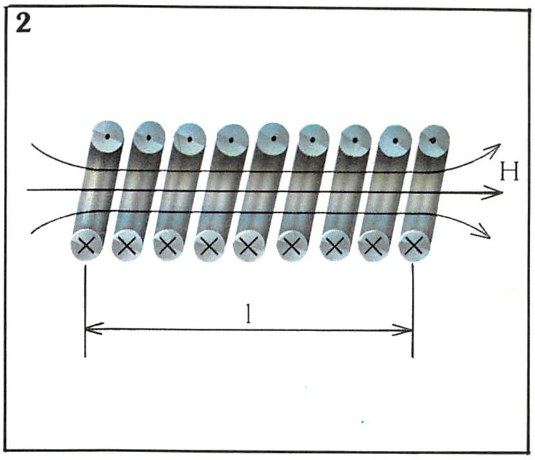
$$H = \frac{NI}{2\pi R}$$

The magnetic field only exists inside the toroid - no magnetic effects are observed outside. This is of great value when a number of coils are used close together as their magnetic fields cannot interact with each other. Toriods are used in high quality transformers and other

1. A single turn coil with current flowing through it produces a magnetic field at its centre.

2. Cross-section of a solenoid.





electrical applications.

Magnetomotive force

If the length of a solenoid is fixed, we can increase the magnetic field by increasing the number of ampere-turns that create it. The number of ampere-turns on which the field depends is called the **magnetomotive force,** abbreviated MMF (this is analogous to electromotive force in electrical circuits) and is given by the symbol F. F is measured in units of ampere-turns (At), thus we have:

$$F = NI = Hl$$

where l is the length of the solenoid; or for a toroidal coil the mean circumference of the toroid.

Magnetic flux

Although the magnetic field is influenced by the shape of the coil it is chiefly determined by the amount of current flowing through it. However, we can increase the magnetic effect by using some ferromagnetic material as the **core** of the coil.

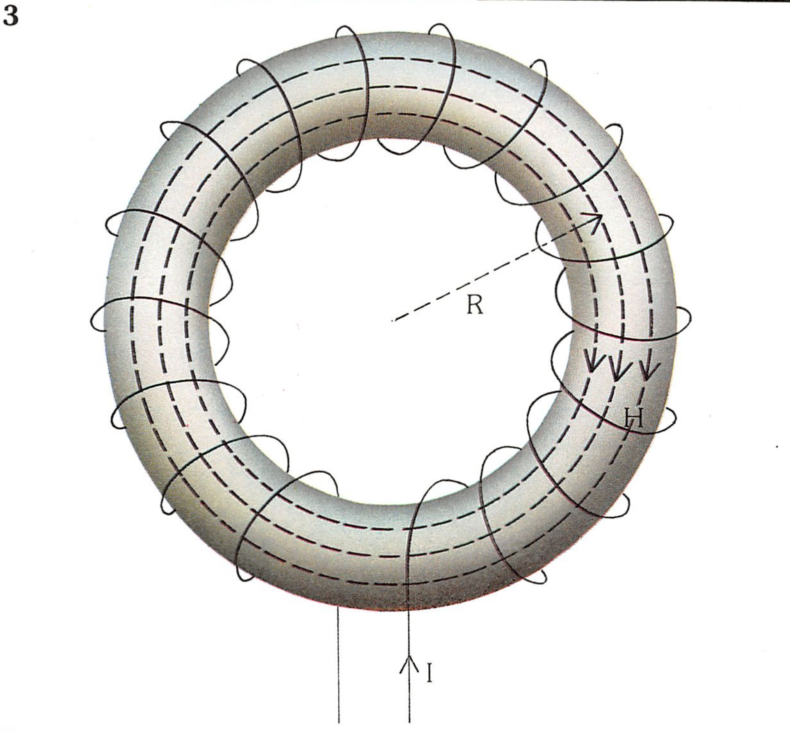
If we connect the solenoid shown in *figure 4* to an electrical supply and pass a current through it, we shall see that it behaves like a magnet. If, when looking at one end of the coil, we see that the current is flowing in a clockwise direction we know that that end of the coil acts as a **south pole** (S). Conversely if the current flows in an anticlockwise direction, that end of the coil will act as the **north pole** (N).

Placing a bar of soft iron inside the coil would increase the strength of the magnetic effect by possibly as much as 1000 times. Thus we have intensified the magnetic field without having to increase the current that causes it. The magnetic effect resulting from the magnetic field is called the **magnetic flux** and is represented by the symbol Φ. It can be thought of in a similar way to the definition of electric flux. Magnetic flux is measured in webers (Wb).

The magnetic flux density (B) is defined as the amount of flux which passes through an area of 1 square metre. Thus, if a total flux Φ is passing through an area of A square metres we shall have a flux density B (measured in webers per square metre - Wbm^{-2}) that is given by:

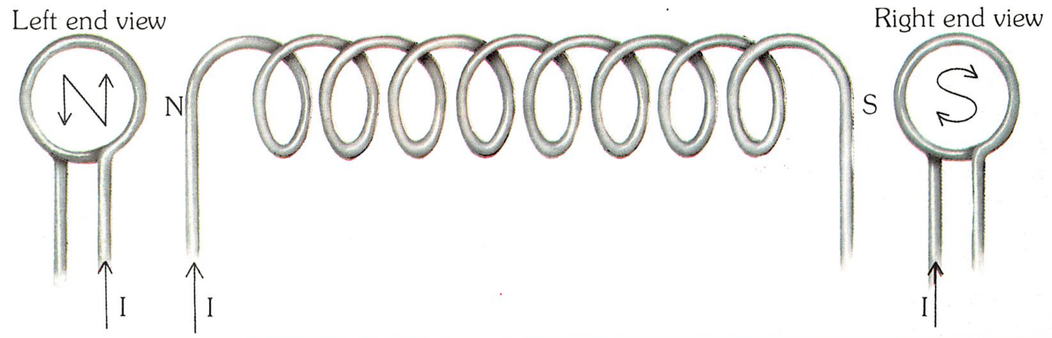
$$B = \frac{\Psi}{A}$$ □

3



3. A toroid is a solenoid bent around into a circular shape.

4



4. A toroid connected to an electrical supply behaves like a magnet.

Optimisation and decoding

Different ways of building the same circuit

In the last chapter we looked at the laws of logic and saw how they could be used to combine and simplify logical expressions. We also began to see how circuits could be built from gates to make or **implement** these expressions.

The implementation of a gate circuit has to take many factors into account as well as the actual logical expression it is to represent. As well as the all important economic considerations, the circuit design must also account for parameters such as the propagation delays involved, power requirements and the overall number and type of gates used.

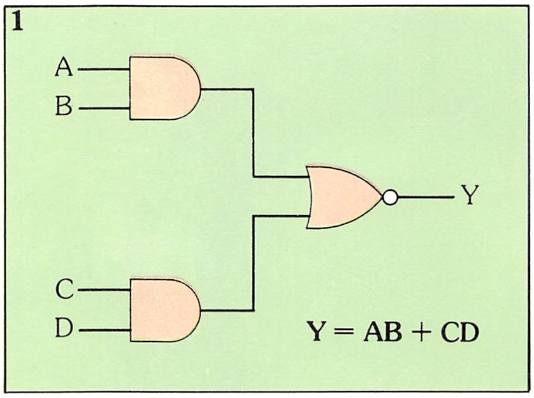
It is often the case that a designer will only have a limited choice of gates available; in the TTL and CMOS series' of devices NAND and NOR gates are the most common. As we mentioned in the last chapter, it is possible to combine these gates to construct all the logic functions and to make gates with larger numbers of inputs than are usually provided.

Similarly, circuit functions can be implemented in different ways. For instance the AND-OR-INVERTER circuit in *figure 1* can also be represented by the circuit in *figure 2*. As an example, let's conside how the following function can be built using two different methods:

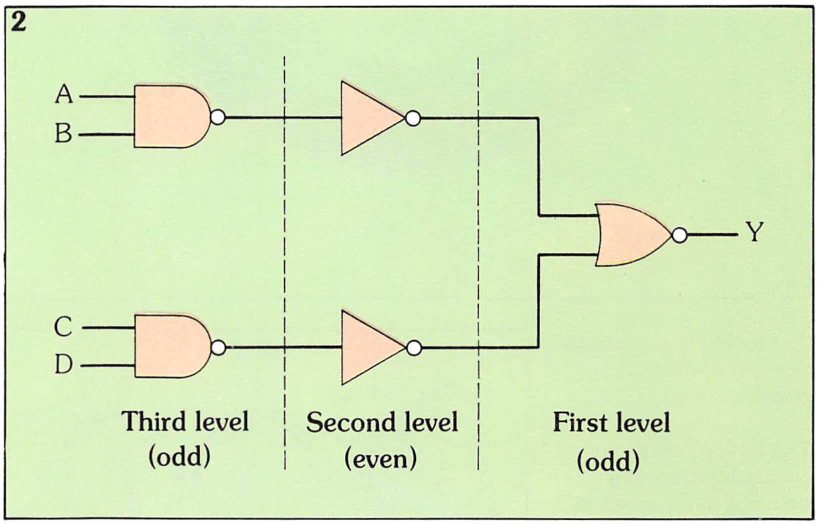
$Y = AB + AC + BC$

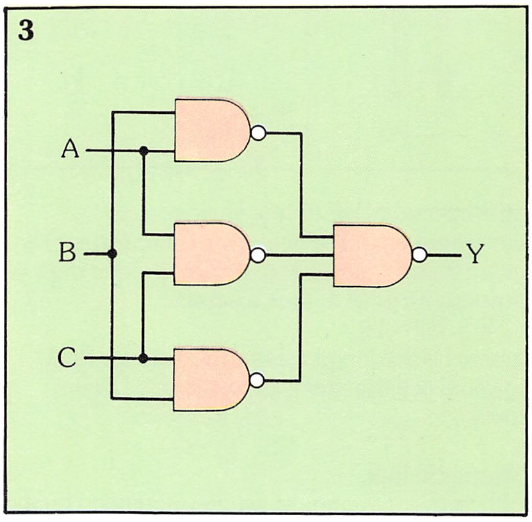
The first uses a NAND gate at the first level so that it performs an OR function. Since there are three terms to be logically added (AB, AC, BC), the gate must have three inputs. However, before being added together these three terms must be multiplied together (A.B, A.C, B.C). This can be done either by a NAND gate on an even level or by a NOR gate on an odd level.

Because of this, there are at least two



1, 2. Different ways of representing the same AND-OR-INVERTER circuit.





3. To build the function Y= AB+AC+BC, a three-input NAND gate on the first level is preceded by three two-input NAND gates on the second level.

possible ways of building this one function. The first uses a three-input NAND gate on the first level, preceded by three two-input NAND gates on the second level (*figure 3*).

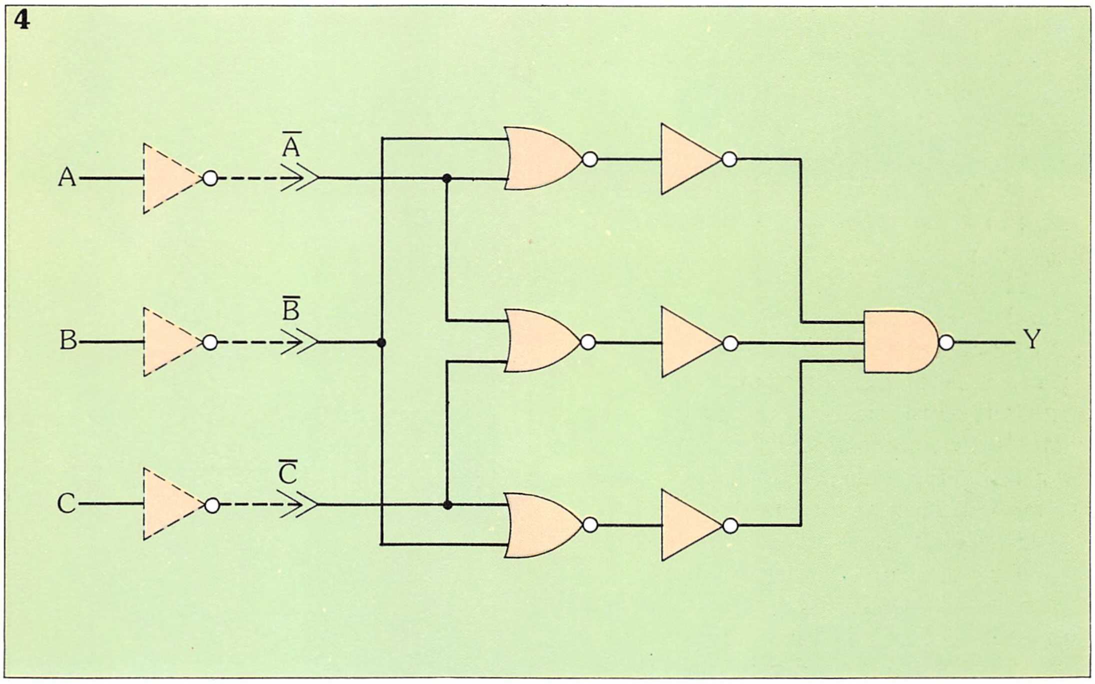
The second uses a three-input NAND gate on the first level with three two-input NOR gates on the third level (*figure 4*). Note that the NOR gates are on the *third* level and not on the second. This means that inverters must be placed on them because a NOR gate on an odd level will only perform an AND function if the inputs are inverted.

By applying Boolean rules, the second method allows the equation to be rewritten as:

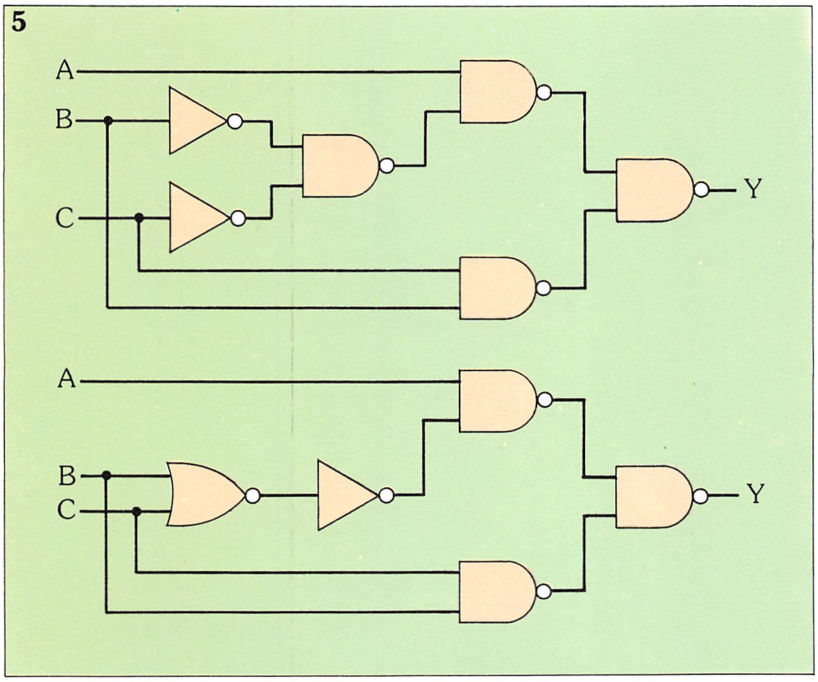
$Y = A(B + C) + BC$

which contains the sum of the two terms. A two-input NAND gate is used on the first level, the product BC is made on the second level using a two-input NAND gate. The term A (B + C) is also obtained by a two-input NAND on the third level, or a NOR on the fourth level. Both of the combinations are shown in *figure 5*.

4. An alternative method of building the function shown in figure 3 is to use a three-input NAND gate on the first level with three two-input NOR gates on the third level.



5. The term A(B + C) is either obtained by a two-input NAND on the third level (above) or a NOR on the fourth level (below).



Expansion of logic gates

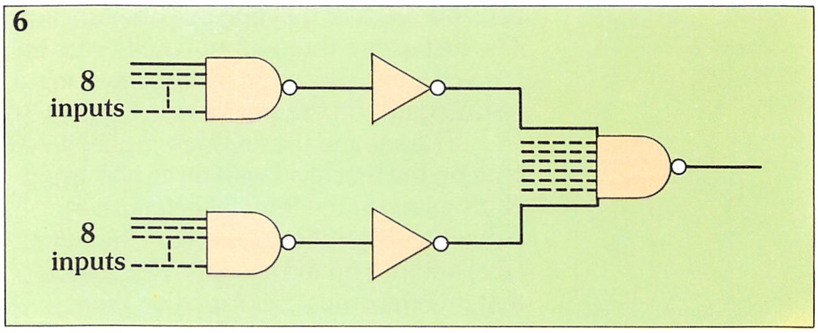
Occasionally a circuit designer will find that a gate with more inputs than are normally available will be needed. There are methods of combining circuits that will give gate equivalents with more than the usual maximum of eight inputs.

NAND circuits

Two, three, four and eight-input NAND gates are available in the TTL 74 series. NAND circuits with more than eight inputs can be built by using a combination of several gates. One way of doing this is shown in *figure 6*. The number of NAND gate inputs is increased by adding AND

circuits made up of NANDs followed by inverters. This method gives NAND circuits with a maximum of 64 inputs – an eight-input inverted NAND connected to each input of the first level NAND.

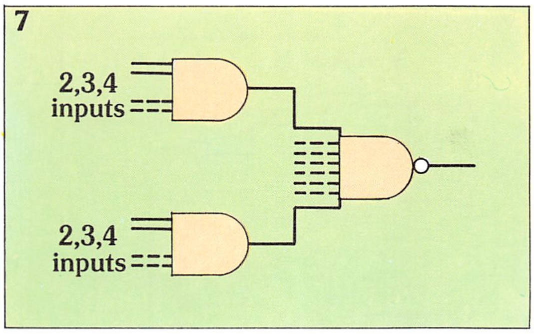
Figure 7 shows an alternative method of building this circuit. Gates from: a 7408 quadruple, two-input AND IC; a 7421 dual, four-input AND IC; a 7411 triple, three-input AND IC, can be used at the

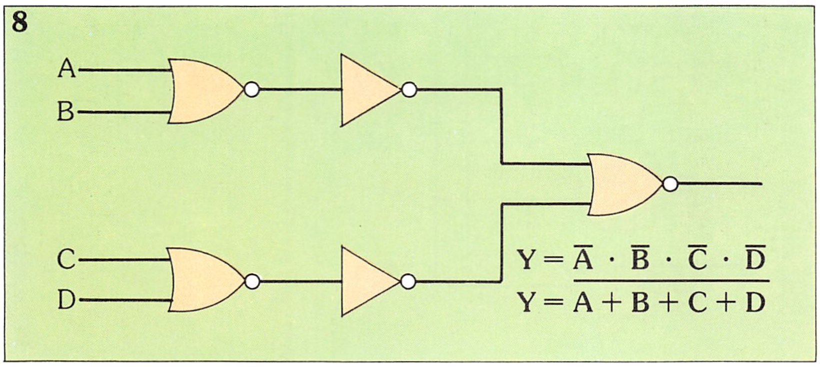


6. A combination of several gates can be used to build a NAND circuit.

7. Alternative method of expanding inputs for the NAND function.

8. A four-input NOR circuit.





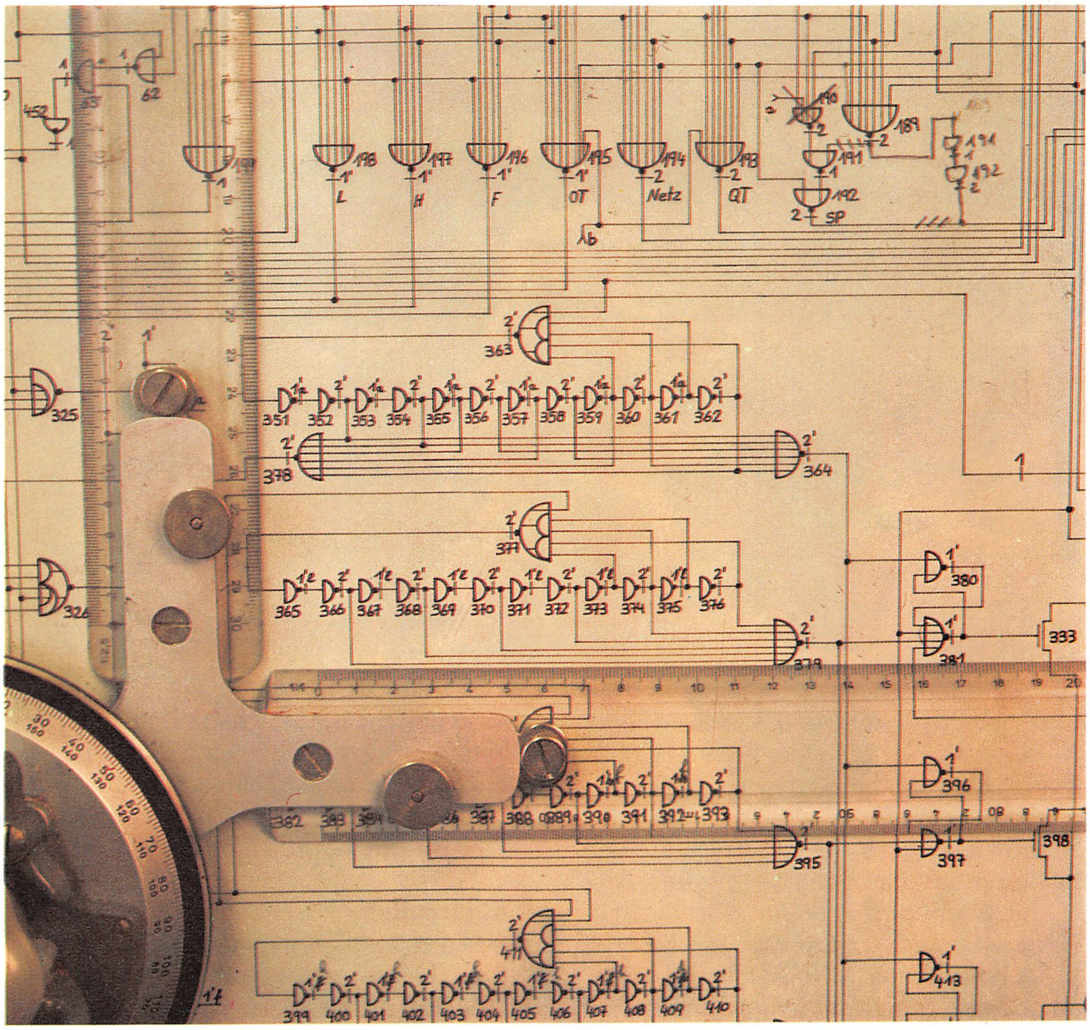
second level. This gives a NAND circuit with up to 32 inputs consisting of an eight-input NAND with a four-input AND connected to each of its inputs.

NOR circuits

Gates from a 7402 quadruple two-input NOR IC can be arranged as shown in *figure 8*, to obtain a four-input NOR gate. Complete AND-OR-INVERTER gates are also available for building NOR functions. To do this only one input terminal of each AND gate must be used (the others must be connected to logic 1, or the terminal used). *Figure 9* shows a gate from a 7450 dual, AND-OR-INVERTER IC connected to gates from 7460 dual, four-input expander ICs giving a circuit, the equivalent of which is a six-input NOR gate.

AND circuits

There are various types of AND gates in



Right: The first step in the design of a new chip is the drawing of its logic functions. (Photo: courtesy ITT).

Left: Examination of the film of a printed circuit board from which the board will later be etched.

the TTL 74 series. For example, the 7408 quadruple, two-input AND IC has totem-pole outputs. The 7409 IC is identical to the 7408 but with open-collector outputs.

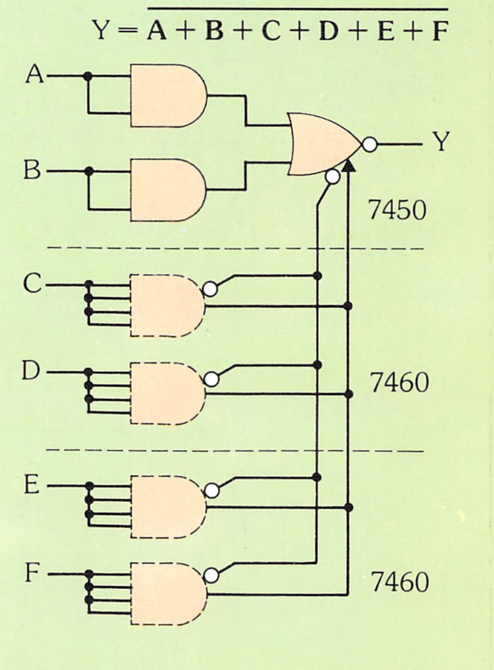
The 7411 is a triple, three-input AND IC, and the 7421 is a double, four-input IC.

Here as well, it is possible to combine gates to obtain AND combinations with more inputs than are initially available.

The number of NAND gate inputs can be increased using the method shown in *figure 10*. The circuit in *figure 11* must be used to do this with AND gates. If gates from the high speed series H are used, AND circuits with up to 32 inputs can be made.

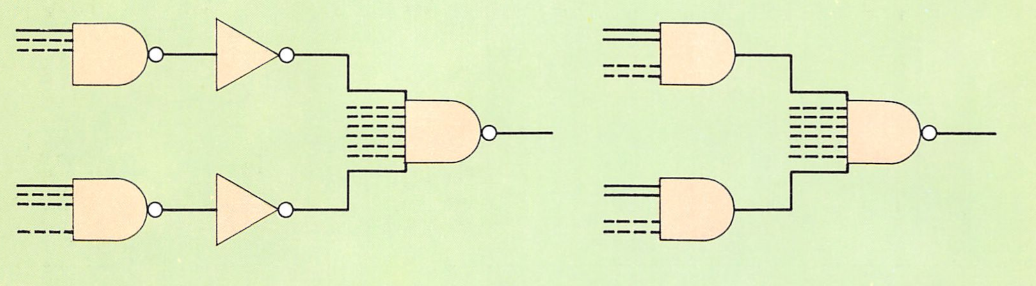
The AND operation can also be obtained by using a gate from a 7402 quadruple two-input NOR IC on the first level and NAND gates on the second level, as shown in *figure 12*. This gives a

9



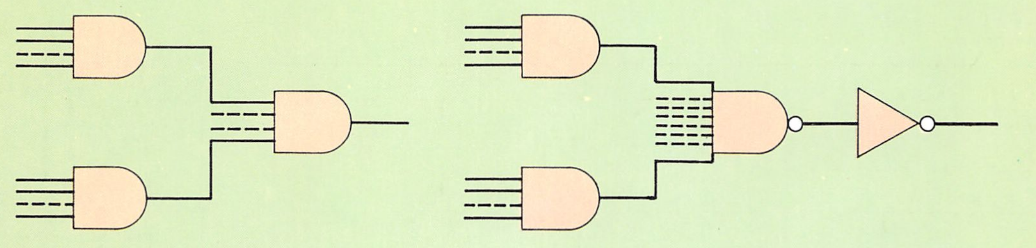
9. Expansion of a NOR gate up to six inputs.

10



10. General method for expanding the number of inputs to a NAND gate.

11



11. General method for expanding the number of inputs to an AND gate.

maximum of 16 inputs. If, instead of the NOR circuit 7402, the previous AND-OR-INVERTER combination is used, the circuits can be modified with expander gates as shown in *figure 13*. This will give a very high number of possible inputs (48). An even higher number of inputs (64) can be obtained by using NOR circuits as shown in *figure 14*. Since a NOR gate placed on an odd inversion level behaves as an AND gate when the input variables are complementary, an AND operation can be performed with NOR circuits capable of supplying the logical product of 6 variables as the output (*figure 15*).

OR circuits

The 74 series TTL includes few ICs containing OR gates; they can however be made from other gates. The simplest

approach is given by the 7402 NOR chip, because inverting a NOR gate's output gives us an OR gate. A different method is shown in *figure 16*. This uses a NAND gate on the first level and a NOR gate on the second. A sixteen-input OR circuit can be obtained in this way.

Remember, a NAND gate on an odd level behaves as an OR gate when the inputs are complementary. With this type of gate, the OR operation can be performed on two, three, four or eight variables. This is shown in *figure 17*.

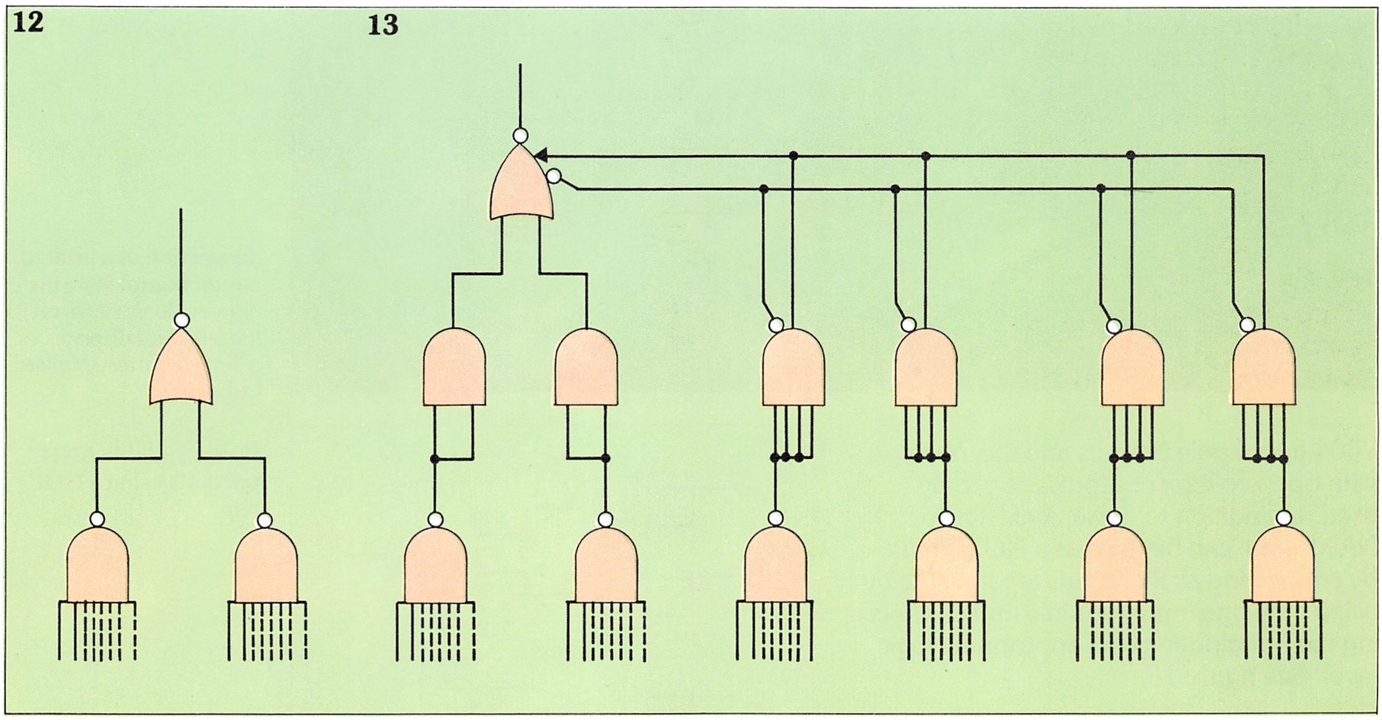
Inverting circuits

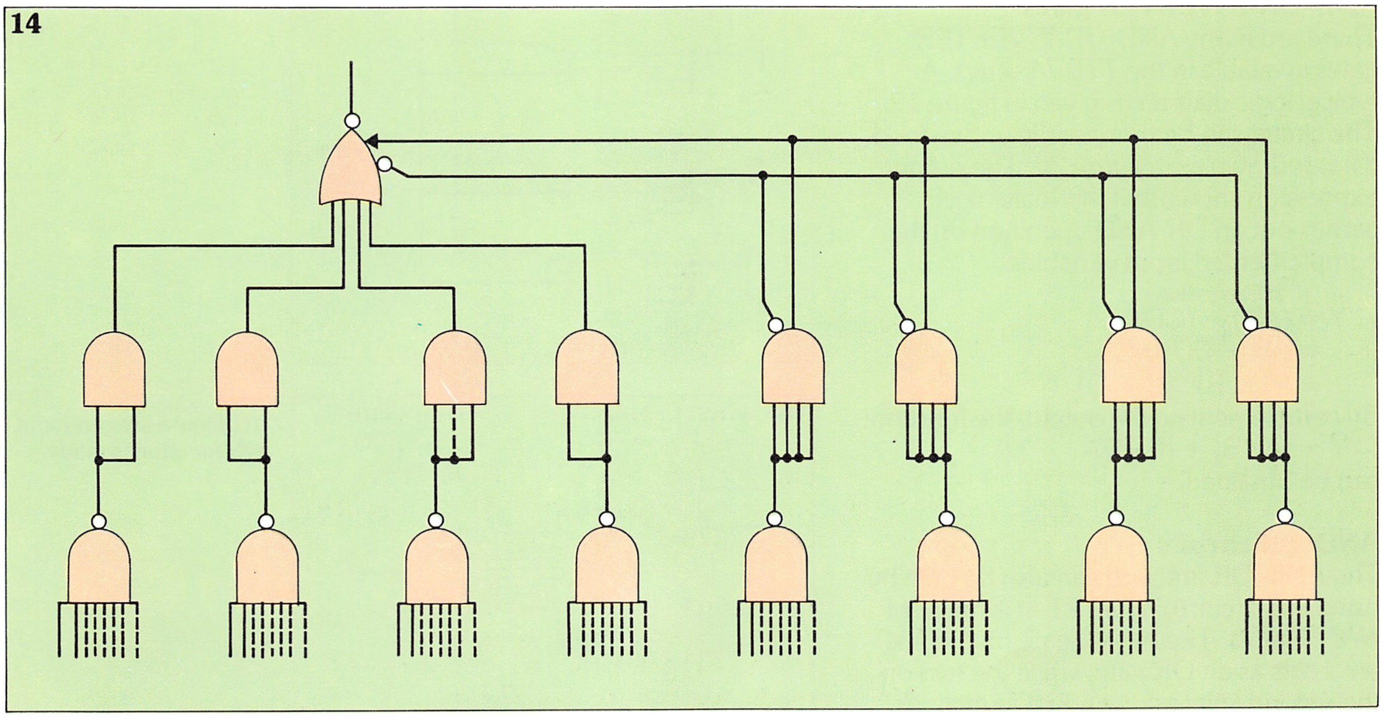
The TTL 74 series includes ICs which perform a NOT operation. They usually contain six single-input NOT gates. The

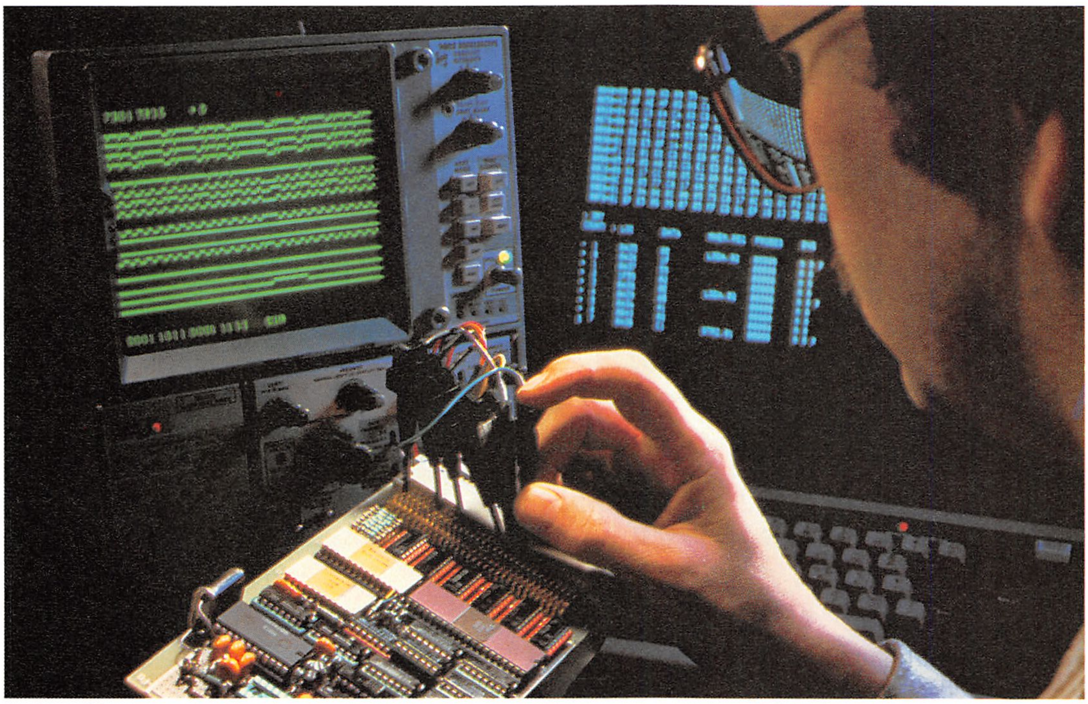
12. Design for a sixteen-input AND gate.

13. Expansion of an AND gate up to 48 inputs using AND-OR-INVERTERS.

14. A 64-input AND gate.







Inspection of a printed circuit board. Note the different logic states on the left hand display. (Photo: courtesy Mullard Ltd).

7404 totem pole outputs and the 7405 with open collector outputs are widely used. In addition to these, any NOR or NAND gate can be used as a NOT circuit by connecting all the inputs together, or by using only one input terminal and connecting the remainder to the appropriate logic level. See *figure 18*.

AND-OR-INVERTER gates

There are many AND-OR-INVERTER gates available in the TTL 74 series. A typical logic diagram is given in *figure 19*. The circuit can be more easily understood if it is redrawn as in *figure 20*. The output expression shows that this logic block carries out an OR-AND operation on the complemented input variables:

$$Y = \overline{\overline{(AB)}} + \overline{\overline{(CD)}}$$
$$= \overline{AB + CD}$$
$$= (\bar{A} + \bar{B}) . (\bar{C} + \bar{D})$$

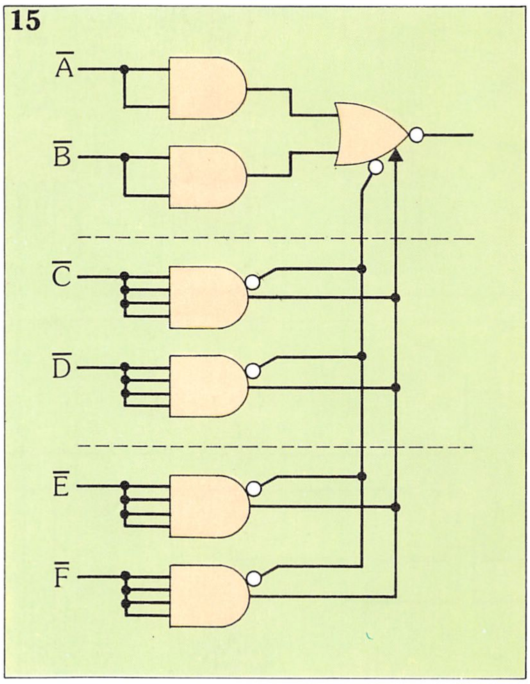
By complementing the output, the function:

$$\bar{Y} = (A . B) + (C . D)$$

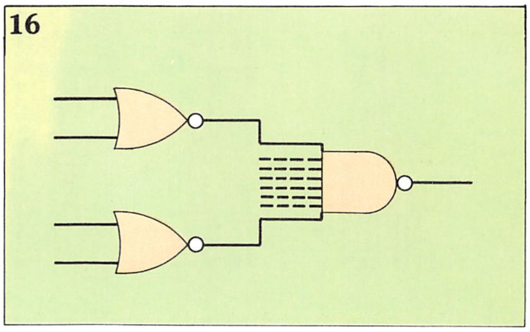
can be obtained.

AND-OR circuits

The AND-OR function can also be built by using the circuit in *figure 21*. It is made of NAND gates. The NAND gate on the first level acts as an OR gate, while the two on the second level perform AND functions.

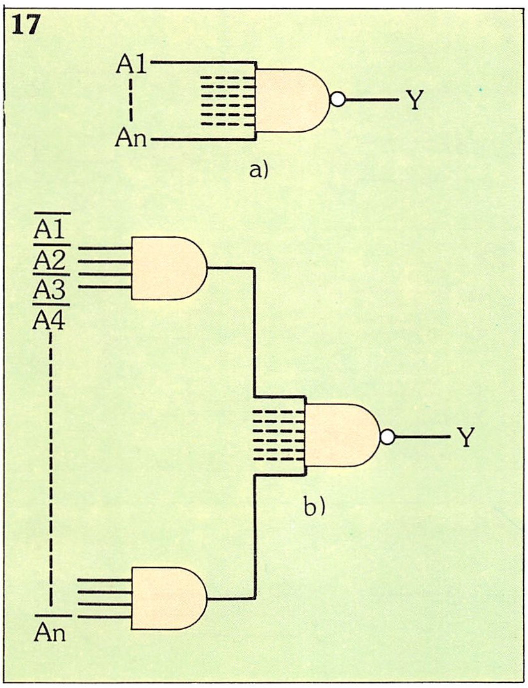


15. Six variable AND functions using a NOR gate.



16. How a sixteen-input OR function is made.

17. Expansion of an OR gate.



The output of the circuit shown is:

$$Y = ABCD + EFGH$$

which analysis shows is given by:

$$Y = \overline{\overline{(ABCD)} \, . \, \overline{(EFGH)}}$$
$$= \overline{\overline{ABCD}} + \overline{\overline{EFGH}}$$
$$= ABCD + EFGH$$

How circuits are expressed in formulae

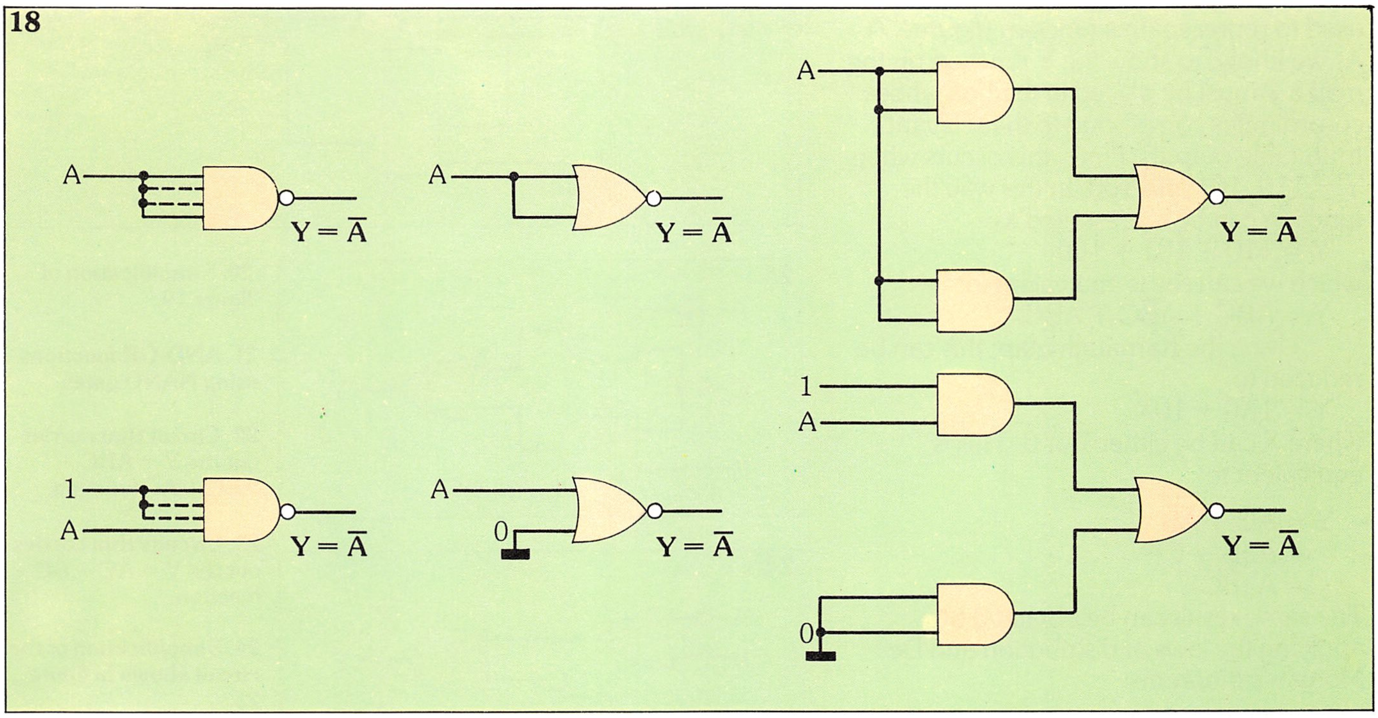
Designing or maintaining logic circuits involves the translation of logical formulae into gate arrangements and vice versa. These functions are known as synthesis and analysis respectively.

Analysing a logic circuit means obtaining a representation of the output function Y. This is done by beginning at the input terminals and working through the circuit to the output, noting the function of each logic gate the signal encounters. In this way a logical expression for the output can be obtained. As an example, look at the circuit in *figure 22*. Working through the circuit we can see that:

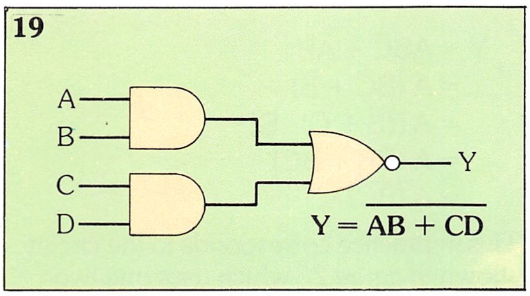
$$Y = \bar{A}BC + AB\bar{C} + A\bar{C}$$

Synthesis, on the other hand, is the process of designing a logic circuit to perform the operations given by a logical expression. Suppose that you wanted a circuit to give the output function:

18. Inverter circuits using standard gates.



19. Logic diagram for an AND-OR-INVERTER.



$Y = \bar{A}C + BC$

The circuit in *figure 23* will perform this function. This is a two-level circuit with two ANDs, an OR and a NOT gate.

The same logic function can also be built using fewer gates. Using the property of association, the equation can be rewritten as:

$Y = C\,(\bar{A} + B)$

The circuit can now be made from an AND gate, an OR gate and a NOT gate, which is a simpler, more economical method than the previous one. This is shown in *figure 24*.

The use of Karnaugh maps is a very common method of simplifying logic circuits by reducing the number of components used. The output function for the circuit in *figure 25* is:

$$Y = AB\bar{C} + A\bar{B}C + A\bar{B}\bar{C}$$

Figure 26 shows this as a truth table. A three-variable Karnaugh map can also be used to represent this function (*figure 27*). As we know, to show the Y function on the map a 1 must be placed in the box whose co-ordinates correspond to the relevant truth table outputs. Here, this occurs when Y = 110, 101 and 100. In this way the function can be represented as:

$Y = 110 + 101 + 100$

which we can see is equivalent to:

$Y = AB\bar{C} + A\bar{B}C + A\bar{B}\bar{C}$

Using the Karnaugh map, this can be reduced to:

$Y = 1X0 + 10X$

where X can be either 1 or 0. This is equivalent to:

$$\begin{aligned} Y &= A\bar{C} + A\bar{B} \\ &= A\,(\bar{B} + \bar{C}) \\ &= A\,(\overline{BC}) \end{aligned}$$

The same result can be obtained by applying the laws of distribution and De Morgan's theorems:

$$\begin{aligned} Y &= AB\bar{C} + A\bar{B}C + A\bar{B}\bar{C} \\ &= A\,[(B\bar{C}) + (\bar{B}C) + (\overline{BC})] \end{aligned}$$

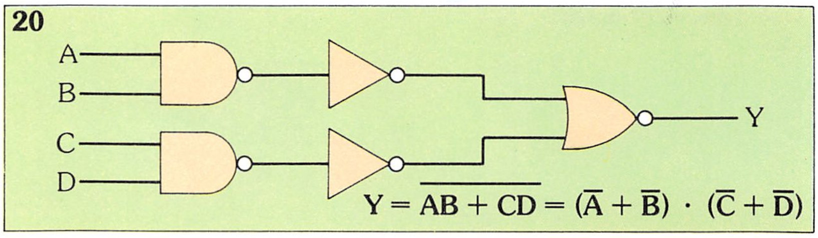
The last two brackets can be simplified to:

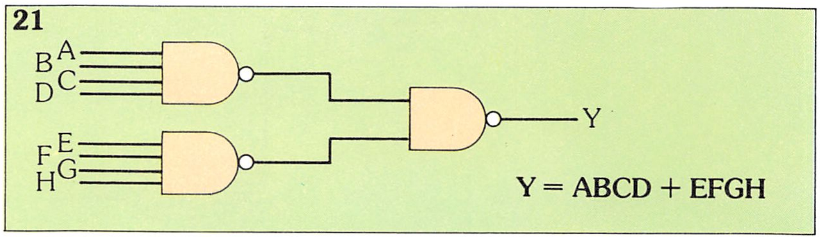
$$\begin{aligned} A\bar{B}C + A\bar{B}\bar{C} &= A\bar{B}\,(C + \bar{C}) \\ &= A\bar{B} \end{aligned}$$

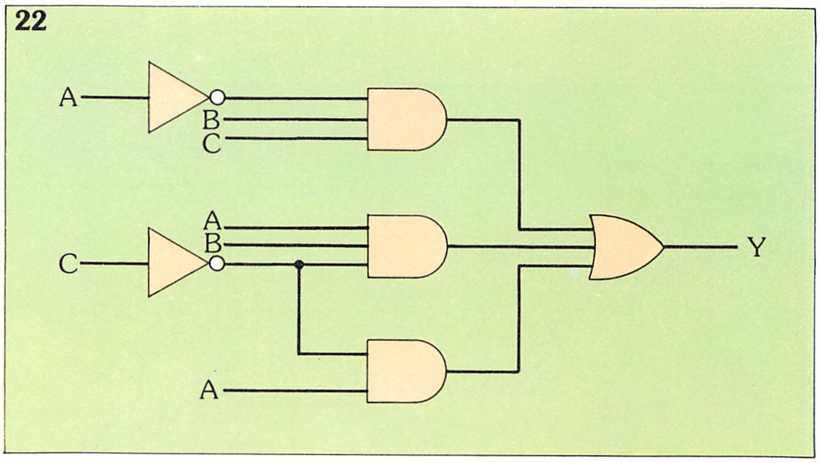
thus:

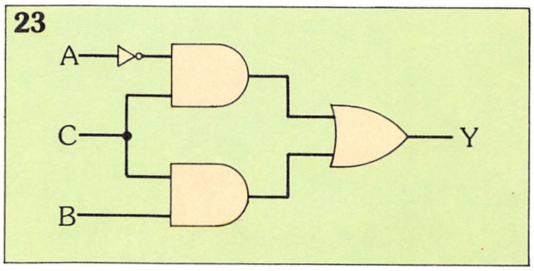
$$Y = AB\bar{C} + A\bar{B}$$

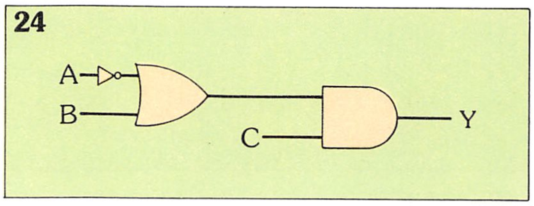
Applying De Morgan's laws we get:











20. Simplification of figure 19.

21. AND-OR functions using NAND gates.

22. Circuit that carries out the $Y = \bar{A}BC + AB\bar{C} + A\bar{C}$ function.

23. Circuits that carries out the $Y = \bar{A}C + BC$ function

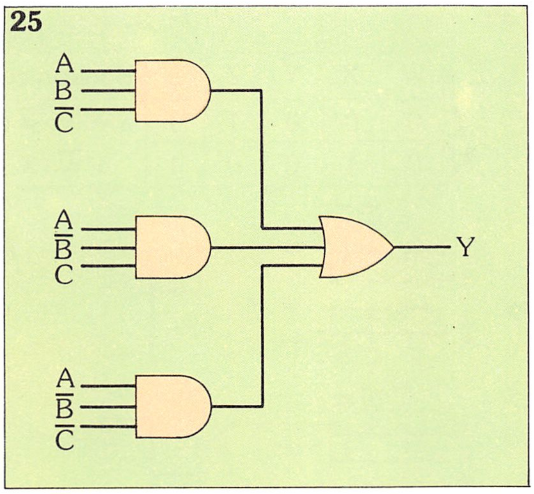
24. Simplification of the circuit shown in figure 23.

$$\begin{aligned} Y &= AB\bar{C} + A\bar{B} \\ &= A\,(B\bar{C} + \bar{B}) \\ &= A\,[(\bar{B} + C)\,.\,B] \\ &= A\,(\bar{B}B + BC) \\ &= A\,(\overline{BC}) \end{aligned}$$

This in practice corresponds to the circuit shown in *figure 28* which uses one two-

25, 26. Circuit and truth table for the function $Y = AB\overline{C} + A\overline{B}C + A\overline{B}\,\overline{C}$.

25



26

Truth table

A	B	C	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

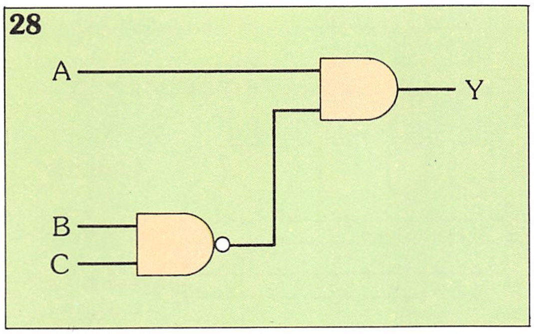
27. A three variable Karnaugh map for the function shown in figure 25.

28. Simplification of the circuit shown in figure 25.

27

C \ AB	00	01	10	11
0	0	0	1	1
1	0	0	1	0

28



input NAND gate and one two-input AND gate. This gives the simplest most economical circuit for this function.

This method of reduction of a function relies on the way that minterms are grouped on the Karnaugh map. Whenever two minterms occupy adjacent squares they may be represented by the logic function of the combined group. For example consider the top two right hand squares in the Karnaugh map (*figure 27*). These represent the terms $AB\overline{C}$ and $A\overline{B}\,\overline{C}$, but can be simplified to $A\overline{C}$, which describes the position of these two squares on the map. Similarly, the minterms that represent $A\overline{B}\,\overline{C}$ and $A\overline{B}C$ occupy vertically adjacent squares on the map and can be simplified to the two- square label $A\overline{B}$. Thus the whole function:

$$AB\overline{C} + A\overline{B}\,\overline{C} + A\overline{B}C$$

is identical to:

$$A\overline{C} + A\overline{B}$$

You will notice that the minterm $A\overline{B}\,\overline{C}$ is represented in both of the above groups but of course this does not affect the final function.

The grouping of functions in this way may be extended for up to four minterms in a line or in a two-by-two block. These may be labelled by a single variable.

Sum-of-products circuit

Figure 29 reproduces the part of the truth table relating to segment **a** of the keyboard encoder. The algebraic function is shown on the right of the table. It is quite simply the 'shorthand' version of the overall formula:

$$a = \overline{W}.\overline{X}.\overline{Y}.Z + \overline{W}.X.\overline{Y}.\overline{Z}$$

Figure 29 illustrates the gate network from this equation. The inverters supply a NOT function for W, X, Y and Z. The two AND gates produce P_1 (logic product 1) and P_4 (logic product 4). The OR gate produces the function $P_1 + P_4$.

As we know, the AND function of various signals is called the **logical product** because it resembles multiplication when written in terms of Boolean algebra. Similarly, the OR function of different signals is called the **logical sum.** Therefore, given that the output **a** is obtained as an OR function of distinct AND functions, it is described as being obtained as a **sum-of-products.** Every circuit composed of AND gates applied to the inputs of OR gates is called a **sum-of-products circuit**.

How the decoder network operates

Figure 30 shows the complete diagram of a seven-segment BCD decoder. As you can see, there is an AND gate for each of the input combinations which produce a 1 on the output. Since the number 8 (represented in binary) at the decoder's input gives an output of all 0s, output C has no AND gate. (Remember, in this circuit an output of 0 indicates on for a segment of the display, while 1 indicates off. See *figure 4* in *Digital Electronics 6* for the truth table of this circuit.)

The input signals for the AND gates are W, X, Y and Z or $\overline{W}$, $\overline{X}$, $\overline{Y}$ and $\overline{Z}$. The complementary inputs are produced by four inverters, shown at the top left in the diagram (*figure 30*).

The outputs of each AND gate are labelled with the symbol P (products). This is an analogy of the function that the gate produces. In Boolean form, for example, it is stated like this:

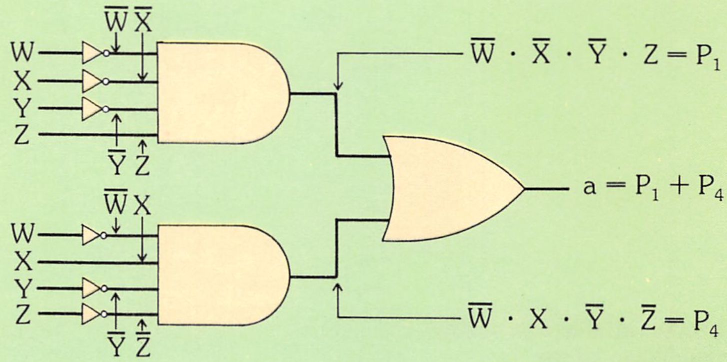
$P_0 = \overline{W}\overline{X}\overline{Y}\overline{Z}$; $P_1 = \overline{W}\overline{X}\overline{Y}Z$

and so on. Output **a** is fed from lines P_1 and P_4 thus **a** is equal to $P_1 + P_4$:

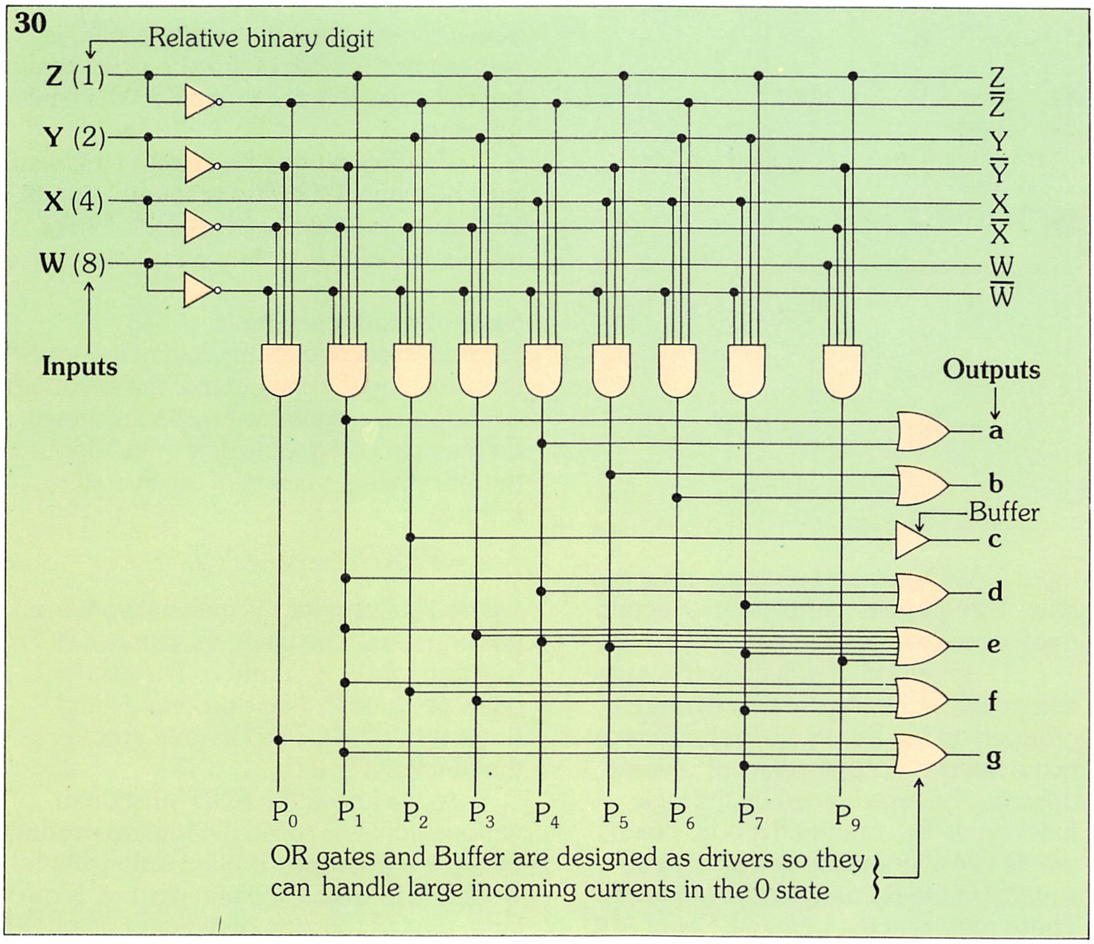
$P_1 + P_4 = \overline{W}\overline{X}\overline{Y}Z + \overline{W}X\overline{Y}\overline{Z}$

29

	W	X	Y	Z	a	Boolean equation for output a	
1	0	0	0	1	1	$a = \overline{W}\cdot\overline{X}\cdot\overline{Y}\cdot Z$	$a = P_1 + P_4$
4	0	1	0	0	1	$+ \overline{W}\cdot X\cdot\overline{Y}\cdot\overline{Z}$	

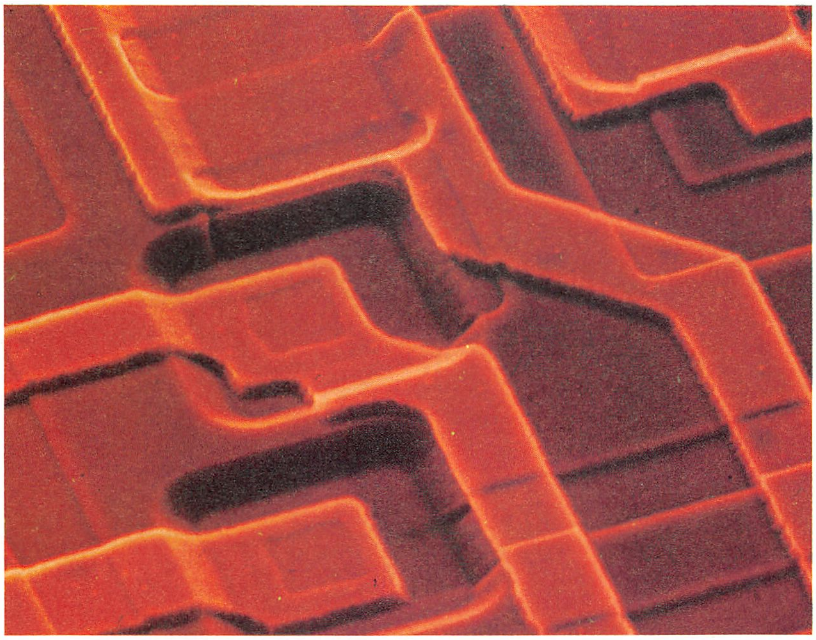


30



29. Truth table and logic associated with segment **a** of the BCD-to-seven-segment decoder.

30. Complete redesign for a BCD-to-seven-segment decoder. The OR gates and buffers are used as drivers.



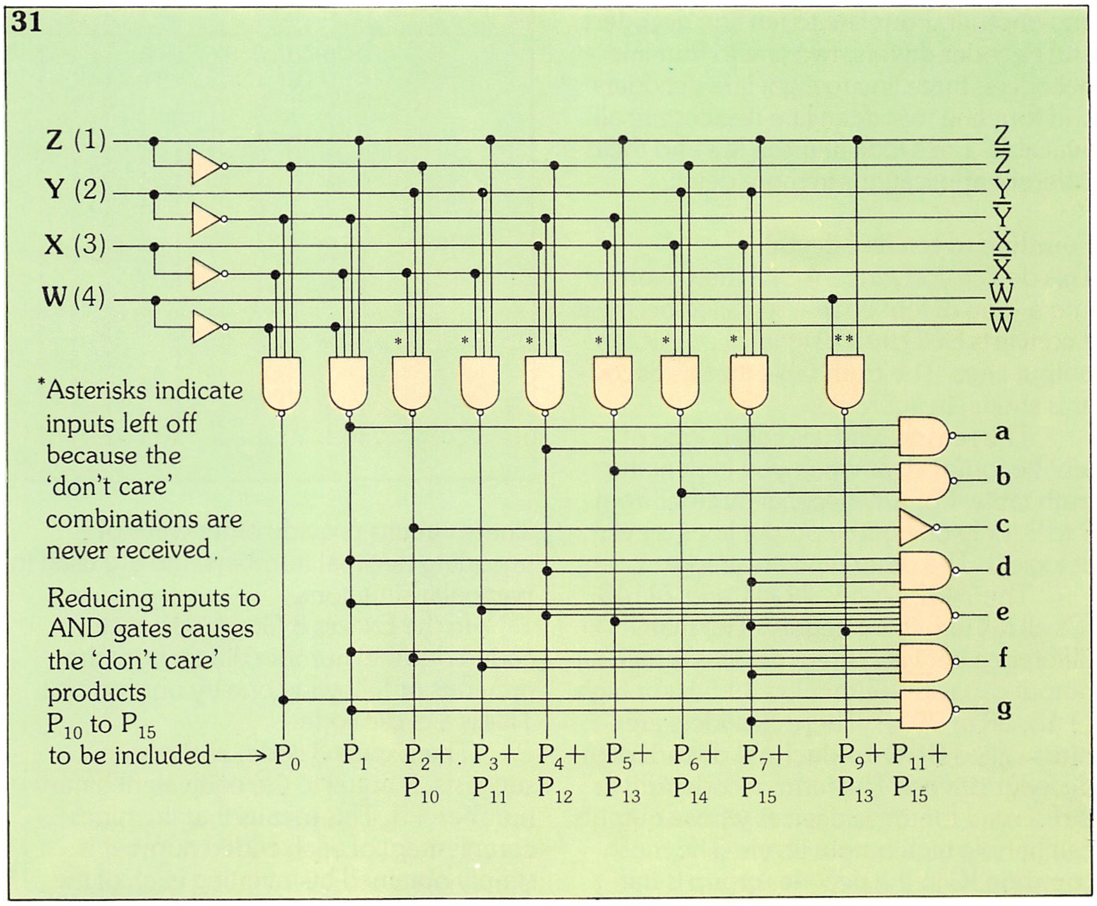
Part of a chip seen through an electron microscope.

Similarly, all other output functions can also be taken directly from the truth table. Output **c** is in fact simply P_2, however it cannot be taken directly from an output line as a buffer has to be placed on the output to protect the AND gate from overloading. This ensures that the digital signal has the power necessary to drive each segment of the display. In this case both the buffer and the OR gates have to be capable of supplying enough current to drive the display.

How the decoder is really built

The decoder circuit that we have examined so far has only been an example used to explain the way different logic circuits work together in a system. In most cases, actual decoder networks have different circuits which use fewer gates and are therefore cheaper to build. The circuit shown in *figure 31*, for example, only uses NAND and NOR gates. You can also see that some of the input connections have been removed. Differences in operation between this circuit and the last would only occur if invalid input combinations were used. These combinations were shown in *figure 4* in the last chapter, but circuit designers assume that these combinations will never be received by the decoder.

31. Practical realization of a BCD-to-seven-segment decoder.



Code converters

Electronic digital circuits use many different codes for their input and output combinations. Just as two people who speak different languages do not understand each other, two circuits which use different codes do not work together properly. We can see that the codes need to be *translated* to suit different equipment, and to do this **code converter** circuits are needed.

A schematic diagram of a code converter circuit is shown in *figure 32*. The type of truth table is shown, but the actual combinations have been omitted to save space. The left hand side contains all the possible input combinations. The relative outputs are on the right of the table. This truth table could have 64 input combinations, but not all of these would be necessarily used in a particular code converter.

The TTL family of logic circuits contains many code converters, which are also known as **decoders** or **decoder drivers.** These offer various configurations enabling the designer to convert nearly any code into another. Four line to ten line decoders and decoder drivers; two line to four line decoders; three line to eight line decoders and four line to sixteen line devices are all available. Let's look at these ICs and their different applications in more detail.

Four line to ten line decoder

This device converts a 4-bit binary word into a 'one of ten' code – so called because it converts BCD into a signal at one of ten output lines. The truth table for this decoder is shown in *figure 33*.

The reason why it is called 'one of ten' becomes clear when you look at the truth table. For any decimal number from 0 to 9, only one of the output lines is ever at logic 0. The other nine are at logic 1.

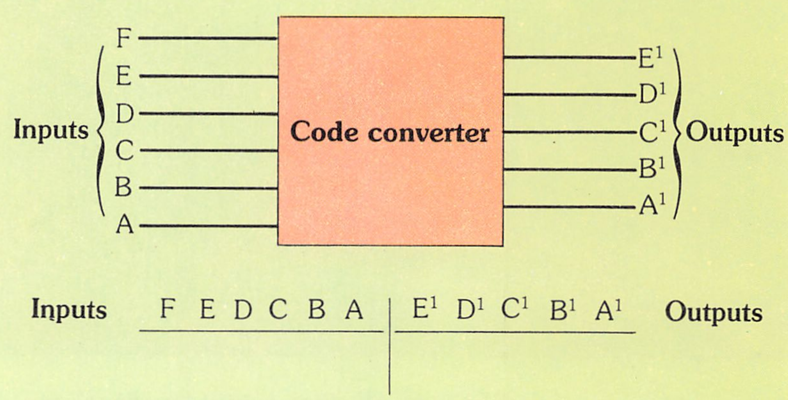
The 7442, 7445, 74141 and 74145 ICs all fall into this category. The main difference between these devices is their output capacity – either low (+5 V) or high (+15, 30 or 70 V). These decoders are often called **BCD-to-decimal decoders** or decoder/drivers. The term decoder/driver is reserved for those devices whose outputs can handle high output levels. The most common IC in the decoder group is the 7442, BCD-to-decimal decoder which is shown in *figure 35*.

The other four line to ten line decoders (7443 and 7444) are known as **Excess-3-to-decimal decoders** and **Excess-3-Gray-to-decimal decoders:** the truth tables of which are shown in *figure 34*.

32. Schematic diagram of a code converter circuit.

33. Truth table for a four-line to ten-line decoder.

32



33

Inputs				Outputs									
D	C	B	A	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1	1	1
0	1	1	0	1	1	1	1	1	1	0	1	1	1
0	1	1	1	1	1	1	1	1	1	1	0	1	1
1	0	0	0	1	1	1	1	1	1	1	1	0	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0
1	0	1	0	1	1	1	1	1	1	1	1	1	1
1	0	1	1	1	1	1	1	1	1	1	1	1	1
1	1	0	0	1	1	1	1	1	1	1	1	1	1
1	1	0	1	1	1	1	1	1	1	1	1	1	1
1	1	1	0	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1

These circuits provide other ways of encoding decimal numbers and are used in particular situations.

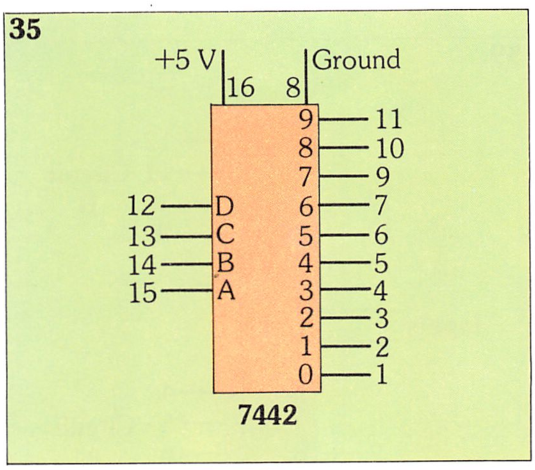
In the Excess 3 Gray code each coded decimal number differs from the previous or following one by only one bit. This is a cyclic code.

The Excess 3 code, as the name suggests, is equal to the equivalent binary number +3. This means that the nines complement of each coded number is simply obtained by inverting each of the

34. Truth tables for the Excess 3 and Excess 3 Gray-to-decimal decoders.

34	Excess 3				Excess 3 Gray				Outputs									
	D	C	B	A	D	C	B	A	0	1	2	3	4	5	6	7	8	9
	0	0	1	1	0	0	1	0	0	1	1	1	1	1	1	1	1	1
	0	1	0	0	0	1	1	0	1	0	1	1	1	1	1	1	1	1
	0	1	0	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1
	0	1	1	0	0	1	0	1	1	1	1	0	1	1	1	1	1	1
	0	1	1	1	0	1	0	0	1	1	1	1	0	1	1	1	1	1
	1	0	0	0	1	1	0	0	1	1	1	1	1	0	1	1	1	1
	1	0	0	1	1	1	0	1	1	1	1	1	1	1	0	1	1	1
	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	0	1	1
	1	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	0	1
	1	1	0	0	1	0	1	0	1	1	1	1	1	1	1	1	1	0
	1	1	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
	1	1	1	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1
	1	1	1	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1
	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
	0	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1
	0	0	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

35. Pin configuration of the 7442 BCD-to-decimal decoder.



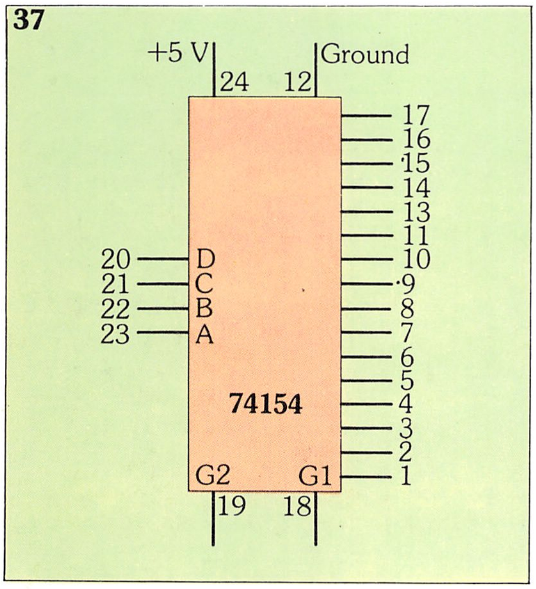
bits. For example, the nines complement of 0110 (3) is 1001 (6). The process of complementing allows arithmetic operations to be performed with ease, and some computers use a similar code to do this.

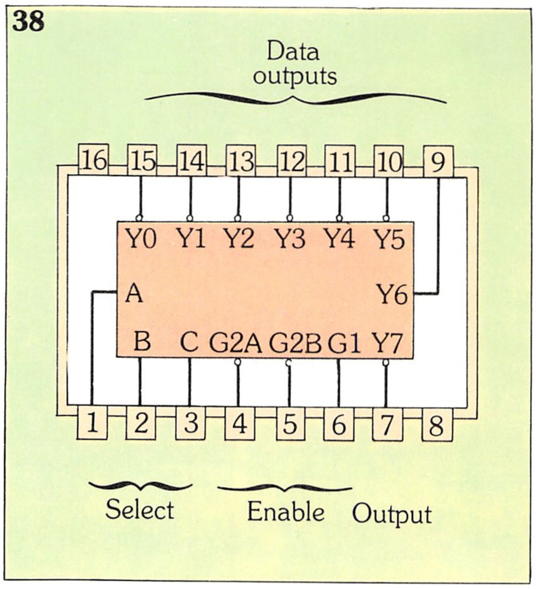
Four line to sixteen line decoder
There are a number of chips specially designed to perform this function. The truth table of the 74154 four line to sixteen line decoder/demultiplexer is shown in *figure 36*, and the chip layout is illustrated in *figure 37*.

The two connections G1 and G2 are

36. Truth table for the 74154 4-line to 16-line decoder.

36 Inputs						Outputs															
G1	G2	D	C	B	A	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	1	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	0	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
0	0	1	0	0	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
0	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
0	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	X	X	X	X	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	X	X	X	X	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	X	X	X	X	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1





37. Pin configuration for the 74154 4 line to 16 line decoder.

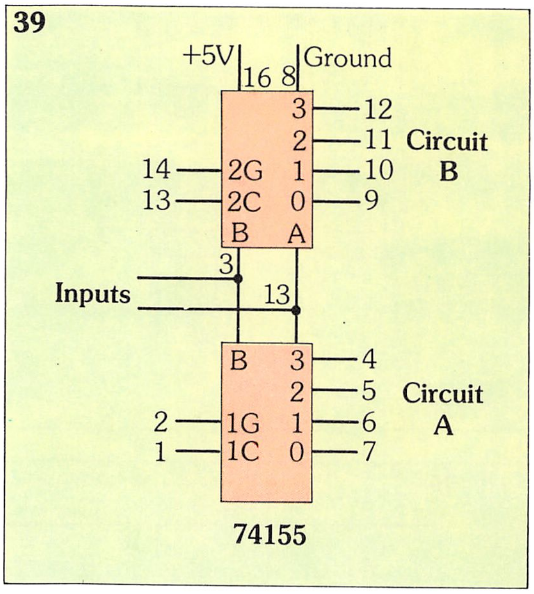
38. Terminal arrangement for the 74138 IC.

strobe inputs, and are important when the chip is used as a demultiplexer. (Demultiplexing is used for sending signals from one set of input wires to a number of different destinations. In this case if four decoders were connected to the input wires, the outputs of decoder 1 would only be active if both G1 and G2 were set to logic 0. Other combinations of G1 and G2 would be used to activate the other three decoders. This will be covered in detail in a later chapter.) To use this IC as a decoder, both of these inputs must be at logic 0.

Three line to eight line decoder
Figure 38 shows the terminal arrangement of the 74138 IC. Although this circuit is catalogued as a three line to eight line decoder/demultiplexer, it is rarely used as the same decoding function can be obtained from the 7442 (BCD-to-decimal decoder) IC, by connecting input D to logic 0 and not using outputs 8 and 9. The 74154 IC can be similarly converted.

Two line to four line decoder
The two line to four line decoder/demultiplexer function is performed by the ICs 74139, 74155 and 74156. Each chip contains two identical circuits. In the 74139 these are completely independent. The 74155 is similar to the 74156 and both have common inputs *(see figure 39)*.

Each single section has two enabling inputs. Circuit A is used as a decoder in this



39. Symbolic representation for the 74155 IC.

case and requires logic 1 at input C1 and 0 at G1. In circuit B, both C1 and G1 have to be set to 0 for it to function.

Under these working conditions, the following truth table is valid for A and B:

Truth table

B	A	0	1	2	3
0	0	0	1	1	1
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	0

(continued in part 10)